Revista Brasileira de Ciências Mecânicas

Journal of the Brazilian Society of Mechanical Sciences

2nd

International Symposium on Spacecraft Ground Control and Data Systems

08-12 February 1999

Foz do Iguaçu - Brazil

PUBLICAÇÃO DA ABCM - ASSOCIAÇÃO BRASILEIRA DE CIÊNCIAS MECÂNICAS

VOL. XXI - SPECIAL ISSUE - 1999

ISSN 0100-7386

REVISTA BRASILEIRA DE CIÊNCIAS MECÂNICAS
JOURNAL OF THE BRAZILIAN SOCIETY OF MECHANICAL SCIENCES
Vol. 1, Nº 1 (1979)-
Rio de Janeiro: Associação Brasileira de Ciências Mecânicas
Trimestral
Inclui referências bibliográficas.
1. Mecânica
ISSN-0100-7386

A REVISTA BRASILEIRA DE CIÊNCIAS MECÂNICAS publica trabalhos que cobrem os vários aspectos da ciência e da tecnologia em Engenharia Mecânica, incluindo interfaces com as Engenharias Civil, Elétrica, Química, Naval, Nuclear, Aeroespacial, Alimentos, Agrícola, Petróleo, Materiais, etc., bem como aplicações da Física e da Matemática à Mecânica.

INDEXED by Applied Mechanics Reviews and Engineering Information, Inc.

Publicação da / Published by
ASSOCIAÇÃO BRASILEIRA DE CIÊNCIAS MECÂNICAS
THE BRAZILIAN SOCIETY OF MECHANICAL SCIENCES

Secretária da ABCM : Ana Lúcia Fróes de Souza
Avenida Rio Branco, 124 18º Andar
20040-001 Rio de Janeiro RJ
Tel.: (021) 221-0438/Fax: (021) 509-7128
www.puc-rio.br/parcerias/abcm
abcm@domain.com.br

Presidente: Carlos Alberto de Almeida
Vice-Presidente: Hans Ingo Weber
Secretário: Paulo Batista Gonçalves
Diretor de Patrimônio: Felipe Bastos de F. Rachid
Secretário Geral: Nestor Alberto Z. Pereira

Secretária da RBCM: Maria de Fátima Alonso de Sousa
UNICAMP - FEM - C.P. 6122
13083-970 Campinas SP
Tel.: (019) 788-3205/Fax: (019) 289-3722
E-Mail: abcm@fem.unicamp.br

JOURNAL OF THE BRAZILIAN SOCIETY OF MECHANICAL SCIENCES
REVISTA BRASILEIRA DE CIÊNCIAS MECÂNICAS

REVISTA FINANCIADA COM RECURSOS DO
Programa de Apoio a Publicações Científicas
MCT CNPq FINEP

JOURNAL OF THE BRAZILIAN SOCIETY OF MECHANICAL SCIENCES
REVISTA BRASILEIRA DE CIÊNCIAS MECÂNICAS

SPECIAL ISSUE:

PROCEEDINGS OF THE 2nd INTERNATIONAL SYMPOSIUM ON SPACECRAFT CONTROL AND DATA SYSTEMS - SCDII

EDITOR FOR THIS SPECIAL ISSUE:

Pawel Rozenfeld
INPE – Instituto Nacional de Pesquisas Espaciais
CRC – Centro de Rastreio e Controle de Satélites
C. P. 515
12201-970 – São José dos Campos, SP – Brazil
Tel.: 55 12 345.6372 – Fax: 55 12 341.1873
E-mail: pawel@beta.ccs.inpe.br

ASSOCIATE EDITORS:

Valcir Orlando
INPE – Instituto Nacional de Pesquisas Espaciais
CRC – Centro de Rastreio e Controle de Satélites
C. P. 515
12201-970 – São José dos Campos, SP – Brazil
Tel.: 55 12 345.6374 – Fax: 55 12 341.1873
E-mail: valcir@gama.ccs.inpe.br

Hélio Koiti Kuga
INPE – Instituto Nacional de Pesquisas Espaciais
DMC – Divisão de Mecânica Espacial e Controle
C. P. 515
12201-970 – São José dos Campos, SP – Brazil
Tel.: 55 12 345.6183 – Fax: 55 12 345.6226
E-mail: hkk@dem.inpe.br

PROGRAM COMMITTEE:

Pawel Rozenfeld – Chairperson	(INPE – Brazil)
Múcio Roberto Dias	(AEB – Brazil)
Rhoda S. Homstein	(NASA – USA)
Hans Peter Piotrowski	(DLR – Germany)
Jean François Kaufeler	(ESOC – ESA)
Veniamin V. Malyshev	(MAI – Russia)
Jean-Claude Terrison	(CNES – France)

The symposium was realized under the auspices of INPE and AEB, and sponsored by:

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 1

ISSN 0100-7386
Printed in Brazil

Foreword

During the five days of SCD II 71 registered participants from 14 countries heard 62 full papers and 8 posters presented by the authors from all over the world.
Because SCD II took place together with 14th International Symposium on Space Flight Dynamics (ISSFD XIV), the participants in one of the symposia were able to attend sessions of their interest in the other one.
The leitmotif of the presented papers was the cost reduction of the space operations. This is, without doubt, the trend of this end of the century when the budget is getting shorter while the missions are getting more involved. The different ways of achieving this goal using the latest technologies and tools is the subject of a great number of papers.
However, the symposium did not limit itself to this subject only. Other very interesting matters were also dealt with. To have a complete picture about what was going on at the symposium you are encouraged to peruse all the papers which constitute this volume.
By publishing these proceedings SCD II have come to its successful end.

Pawel Rozenfeld
Chairperson

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 2

ISSN 0100-7386
Printed in Brazil

Ground Systems I

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 3-9

ISSN 0100-7386
Printed in Brazil

Datalynx^{tm}: A Highly Automated Commercial Satellite Command, Control and Communications Service Network

Trevor C. Sorensen
Dean Bakeris
Todd Probert
AlliedSignal Technical Services Corporation
7000 Columbia Gateway Dr.
Columbia MD 21046 USA
trevor_sorensen@alliedsignal.com

Abstract

AlliedSignal Technical Services Corporation (ASTC) is developing DataLynx™, a commercial worldwide spacecraft tracking, data collection and control system that is due to be operational in mid-2000. DataLynx™ will provide total spacecraft asset management, including mission planning and scheduling; flight control; ground tracking; spacecraft monitoring, trending and engineering analysis; and payload data capture, processing and distribution. DataLynx™ includes a worldwide network of high latitude, high data-rate tracking, telemetry and commanding autonomous ground stations (TAGS) to support remote sensing satellites in polar orbits, and smaller (5-meter) TAGS at lower latitudes to support science missions and rocket launches. The operations of the TAGS are autonomous, with remote monitoring and control from the DataLynx™ Operations Center (DOC), which will be highly automated to reduce risk and cost of operations. DataLynx™ offers a low risk and low cost service for satellite owners who do not want to spend the money and resources to develop their own ground segment infrastructure.
Keywords: *satellite control network, ground stations, spacecraft/mission operations, ground segment*

Introduction

ATSC has entered the rapidly expanding commercial space market. The ATSC approach to this market is based on our traditional support in satellite ground segments including tracking commanding, telemetry collection, data management, and mission operations. The ATSC plan capitalizes upon the growing number of ventures that need ground services, but which individually cannot fully utilize the capacity of dedicated facilities. There is no need for owners of satellites to make huge investments in capital and resources to build and operate satellite ground networks. Instead, they can focus their energies on their core business. ATSC proposes to meet the needs of many such customers by linking a modular, expandable network of facilities located in key geographical locations and offering utility-like services from those facilities. Reliable and low risk satellite asset management and flight operations, and fast and dependable data transmission, will be available on a buy-by-the-use basis. Our approach also takes advantage of ATSC's experience in this field. For more than 45 years, ATSC has designed, built, operated, maintained, and upgraded ground tracking stations. We have been involved in more than 700 spacecraft missions and have established a track record of delivering ground support systems that work, are delivered on time, and are within budget.

An analysis was performed to determine spacecraft customer needs in tracking, data acquisition and commanding (TDAC) services. We discovered that ground segment requirements can be parsed into six distinct natural groupings, based upon band and bandwidth needs. Four of these groupings: deep space science missions (low-rate S-band); near-space science missions (high-rate X/S-band); geostationary satellites; and LEO communication constellations, are not especially suited for a commercial network service because of requirements for either large aperture antennas or a dedicated network. That leaves two groupings that are especially suitable candidates for the service being offered by ATSC:

High data rate X-band systems are generally remote sensing spacecraft in polar orbits. The ground stations required are generally relatively expensive and underutilized.

Low data rate S-band systems are generally low earth orbiting government or university science missions. They are currently serviced by dedicated ground stations or by the 26-meter sub-net of the DSN, which is vastly larger than required to support the missions, but is used because it is available. These missions are of the "faster, cheaper, better" paradigm and are actively seeking less costly, less bureaucratic solutions to their ground support services needs.

Drawing directly from this analysis and leveraging our core competencies, ATSC intends to meet the needs of the of these two groupings, as well as launch providers, by building, owning, and operating a

private satellite control facility and ground tracking network. Collectively, this system is called DataLynx™.

Services Offered to Spacecraft Customers

DataLynx™ is designed to support the requirements of its anchor customer, Space Technology Development Corporation (STDC), which will be launching a remote sensing satellite called NEMO (Naval EarthMap Observer), in mid-2000, plus additional customers with minimal modifications. The DataLynx™ system provides the following services to satellite customers:

- Tracking, telemetry and command (TT&C) of multiple spacecraft
  - Supports X-band (down) and S-band (up/down) initially
  - Metric tracking (initially autotrack angles and eventually two-way Doppler)
- Spacecraft asset management and operations (SAMO) of multiple spacecraft
  - Mission design and preparation (participate in working groups and reviews; support spacecraft I&T; conduct simulations and rehearsals; develop operations procedures)
  - Mission planning and scheduling
  - Flight dynamics (orbit determination/analysis, maneuver planning and execution)
  - Real-time flight operations (command and control, anomaly detection and recovery, management of onboard spacecraft assets)
  - Spacecraft state-of-health data analysis and archival storage
- Data collection, processing, delivery and archival storage
  - Data rates initially up to 300 Mbps (X-band) and 15 Mbps (S-band)
  - Delivery via T1 line or tape using commercial carrier
  - Level 0 data processing (optional)
  - One day or more of telemetry data stored at the ground station

Pricing is similar to standard telecommunications services. Customers pay an activation fee, which covers the cost of preparing DataLynx™ to meet their needs, including all pre-launch operations engineering development. A small monthly fee for maintenance on the system is also charged. The bulk of customer expenditures are during operations after launch. The usage fee is determined based on a customer-selected priority and telecommunications needs. For SAMO customers only a monthly fee is charged after launch in addition to the initial activation fee.

DataLynx™ System Architecture

ATSC designs, builds, and operates each of the proposed DataLynx™ elements for a wide variety of customers and has been doing so for many years. The architecture of the DataLynx™ system draws upon this heritage for both the design and design methodology.

The ground segment architecture, shown in Fig. 1, is comprised of two major engineering elements, the Ground Network System (GNS) and the DataLynx™ Operations Center (DOC). The GNS provides the TDAC services of DataLynx™. There are two basic types of ground stations: TAGS and EGS. The DataLynx™-owned stations are called TT&C Autonomous Ground Stations (TAGS) and are designed either to support high data rate spacecraft (using 11-meter antennas) or low-data rate spacecraft (using 5-meter antennas). The high data rate antennas support mostly the remote sensing missions in polar orbits, while the low data rate antennas mostly support space science missions and launch vehicles. The TAGS stations are supplemented by other existing external ground stations (EGS) that have excess capacity for lease. All the ground stations are nodes on the ground network that provide satellite tracking, uplink commands, and receive both payload and state of health (SOH) telemetry from the spacecraft.

Initial processing of the payload data up to Level 0 can be done at the stations before the data are shipped by tape, Satcom link or land lines to the customer's facility or the DOC. The spacecraft SOH data are sent by land line to the DOC, where the monitor and control of the spacecraft and of the GNS are conducted. Tasking from the customer is received at the DOC and incorporated into the mission planning and sequencing process, which involves the customer in any schedule conflict resolution. The customer receives mission data, which consists of periodic reports, access to real-time mission/spacecraft status, and the SOH data received from the spacecraft. The SOH data are also archived at the DOC for the duration of the contract with the customer. In addition, the TAGS are capable of collecting metric tracking data in the form of angles and Doppler data used to determine a satellite orbit.

Figure 2 shows the high level functional block diagram of the total DataLynx™ system, which also includes the business management function (program management, operations accounting, customer billing, contracts, marketing, and administration of the system). The technical functions, GNS and DOC system, are described in detail.

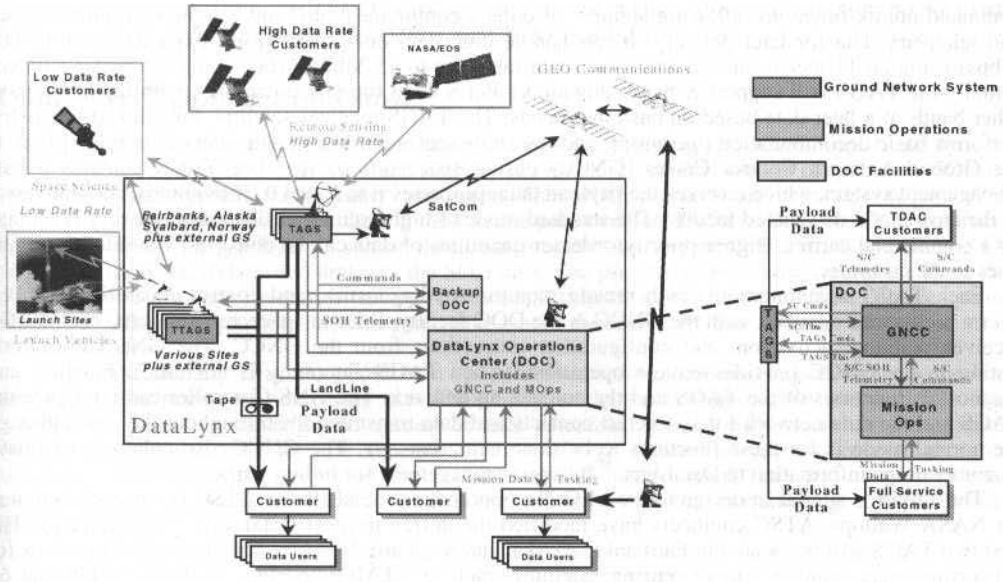


Fig. 1 DataLynx System Architecture

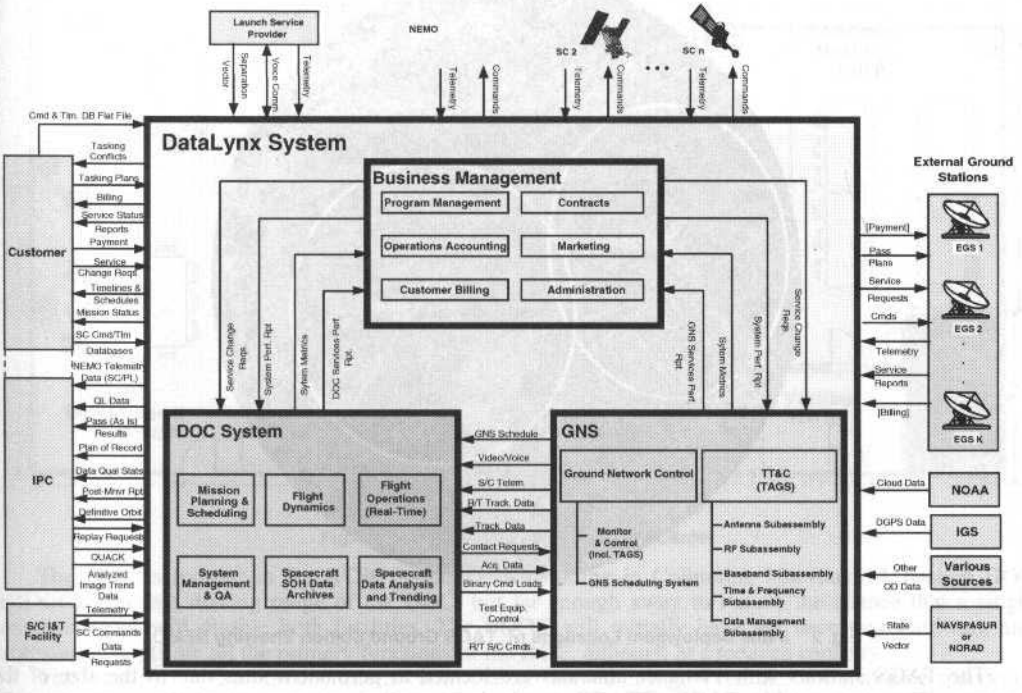


Fig. 2 DataLynx Major Functional Elements

Ground Network System

The GNS consists of two major subsystems: TT&C and the Ground Network Control. The TT&C subsystem consists of the TAGS and any EGS. The TAGS performs the spacecraft data capture and

command uplink functions, allowing simplex or duplex communications, and can store command loads and telemetry data for later delivery. It can handle high data rates (300 Mbps, expandable up to 600 Mbps) using an 11-meter antenna and lower data rates (up to 15 Mbps) when using a 5-meter antenna. Initially the TAGS will support X-band (downlink) and S-band (up-and downlink), with the addition of other bands at a later date based on customer needs. The TAGS receives satellite downlinked telemetry, performs basic decommutation operations, and transmits spacecraft health and safety data in real-time to the Ground Network Control Center (GNCC) during spacecraft passes. The TAGS contains a data management system, which receives the payload data, processes it to Level 0 (if required), and delivers it to the customer's designated facility. The standard mode of high-volume payload data delivery is on tape by a commercial carrier. Higher priority or lesser quantities of data can be delivered via Satcom or land lines to the customer.

Each TAGS is autonomous, with remote monitoring, diagnostics, and control capability provided over a high-speed data link with the GNCC in the DOC facility. Prior to spacecraft supports, each TAGS receives pointing, activation, and configuration instructions from the GNCC. The GNS Controller's station in the GNCC provides remote operations (when TAGS autonomy is not used), monitor, and diagnostics functions of the TAGS and the connecting network. The GNS Controller can configure the TAGS and the data network for spacecraft contacts and data transmissions throughout the day, although the normal mode is for these functions to be done autonomously. The GNCC also collects and relays customer usage information to DataLynx™ business management for billing purposes.

The TAGS is similar in design to the LEO-T autonomous ground station ATSC is currently building for NASA Wallops. ATSC engineers have modified the design to meet DataLynx™ requirements. The first two TAGS will be located in Fairbanks, Alaska, and Svalbard, Norway, which are ideal locations for supporting polar-orbiting remote sensing satellites, such as NEMO (see Fig. 3). Each TAGS can by operated locally for special critical operations.

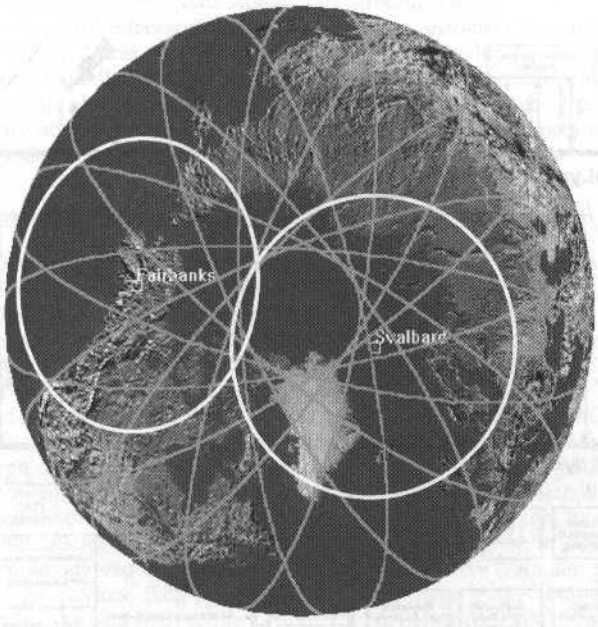


Fig. 2 Initial Deployment Locations of TAGS Ground Station Showing NEMO

The TAGS stations with 11-meter antennas are located at permanent sites due to the size of the antenna. The transportable TAGS (TTAGS), with their smaller antennas, store the electronics and computer equipment in modified trailers or containers that can be transported to different locations to satisfy other customer requirements. This provides flexibility to the GNS and the service that DataLynx™ can offer to customers. These smaller stations are especially suited to support low data rate science missions or to track launch vehicles.

The high latitude TAGS sites increase coverage for the downloading of data from polar orbiting satellites. As remote imaging demand grows, increasing the GNS capacity is simply a matter of

increasing the number of nodes (TT&C sites) on the network, adding additional capability to the existing TAGS, or leasing time on compatible international TT&C sites (EGS).

DataLynx™ Operations Center

The DOC system performs six main functions: mission planning and scheduling, flight dynamics, real-time flight operations, spacecraft data analysis and trending, archival storage of spacecraft SOH data, and system management and quality assurance. The DOC prepares all necessary sequences and commands to operate a spacecraft. During the satellite pass over the TAGS or EGS, the DOC controls the spacecraft, dump the payload or sensor data, and monitor the spacecraft health. Post-pass the DOC processes, analyzes, trends, and archives the SOH data. The primary external interface to the DOC is the customer, who provides spacecraft tasking requests and helps resolve tasking conflicts. Besides periodic reports, DataLynx™ provides an electronic site where the customer can monitor the spacecraft and mission status.

The DOC facility (Fig. 4) contains a spacecraft operations center that is designed to be highly automated. The main real-time string uses Unix-based workstations supported by standard PCs. These workstations have the capability to operate several automated ground stations and provide graphical displays for simultaneous satellite supports. This hardware architecture allows increasingly greater automation with the goal of reducing staffing costs. ATSC has designed, built and maintained control centers for many customers including NASA, the military, State transportation agencies, and foreign governments.

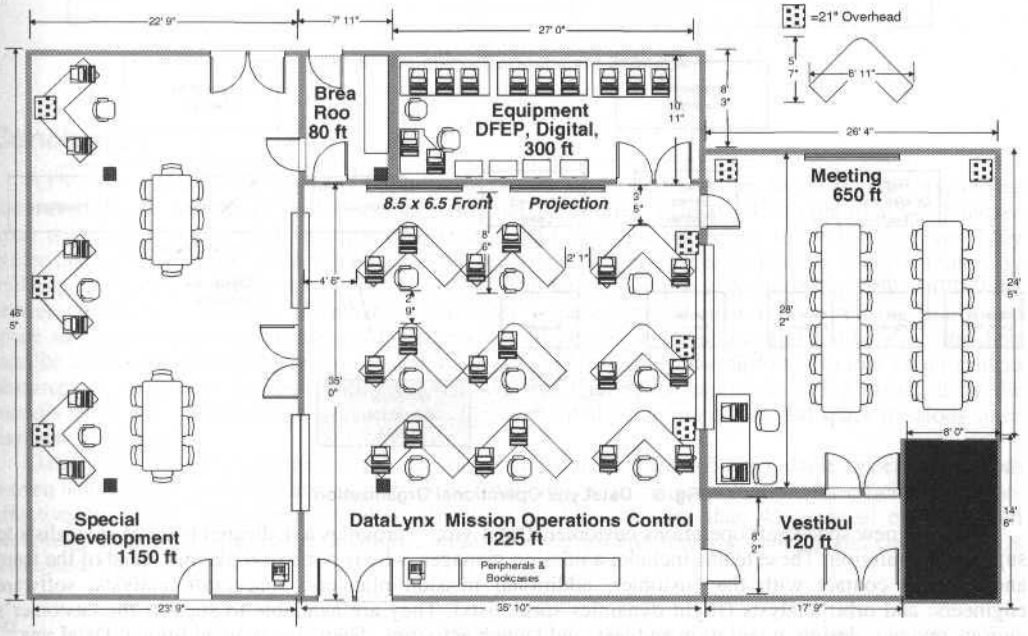


Fig. 4 DataLynx Operations Control Center

The DOC is located in the ATSC headquarters building in Columbia, Maryland. A backup DOC (BDOC) is within driving range of the DOC, but far enough away to reduce the chance that a single external event will disable both facilities. The BDOC will initially have the necessary hardware and software to duplicate all the primary functions of the DOC, although at a reduced capacity.

The DataLynx™ mission operations control center shown in Fig. 4 contains the real-time spacecraft control and the GNS control functions. During the early operational phases, the mission planning and spacecraft analysis functions will also be performed in the control center at consoles that will eventually be available to support real-time operations as the customer base expands. Behind the control center room is an equipment room, which contains the front-end processors, routers, LAN servers, communications and other equipment needed to support the control center. Adjacent to the control center and separated by doors and large glass windows is the special operations room. This provides additional consoles and meeting tables to support special customer operations, launch or integration support, system

development, and training. On the other side of the control center is a large conference room, also with viewing windows. Supporting these operations rooms are several offices and cubicles for the DataLynx™ support staff.

DataLynx™ Operations

The DataLynx™ operational concept calls for continuous operation in the DOC, 24 hours a day for every day of the year. There are two console positions that are staffed on a continuous basis - the Spacecraft Controller, who controls the space assets, and the GNS Controller, who controls the ground assets. The actual number of controllers on duty at any given time depends on the number of spacecraft and ground stations being controlled by DataLynx™. There are a number of positions supporting the real-time operations, but because of the high level of automation of the DataLynx™ system, only these two positions require continual staffing. The basic shift cycle for DataLynx™ operations is two 12-hour shifts a day. The DataLynx™ organization for operations is shown in Fig. 5.

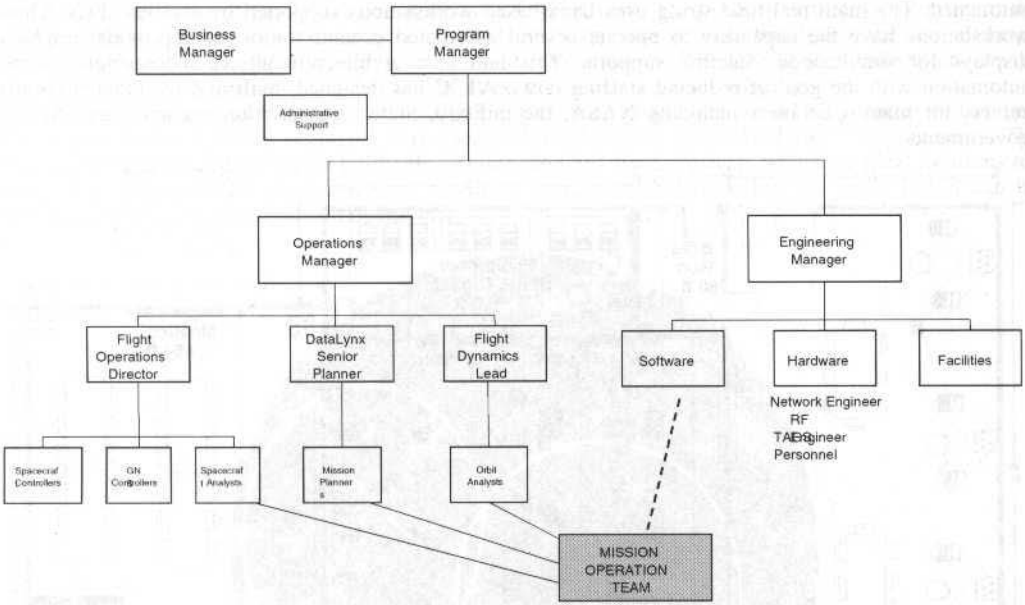


Fig. 5 DataLynx Operational Organization

For each new spacecraft operations customer, DataLynx™ provides a dedicated team of specialists to support that mission. These teams include: a mission manager, who is a mission planner, head of the team and primary contact with the customer; additional mission planners; spacecraft analysts; software engineers; and orbit analysts (flight dynamics specialists). They are available to support the customer's various reviews, design, integration and test, and launch activities. They also train additional DataLynx™ personnel for supporting the customer's mission operations.

Program Schedule

The development of DataLynx™ is being planned (Fig 6) to ensure that the system will be ready to support the launch of the first customer, NEMO, which is currently scheduled for mid-2000.

One driver for the schedule is the desirability of installing the antennas for the TAGS sites at high latitudes during warmer weather (June through September). Each site, including antenna mount, has to be prepared before hand to receive the antenna. Delivery of the antenna to Alaska is scheduled for July, 1999; the DOC will be completed during the fall of 1999; and the DataLynx™ system acceptance testing is scheduled for March, 2000.

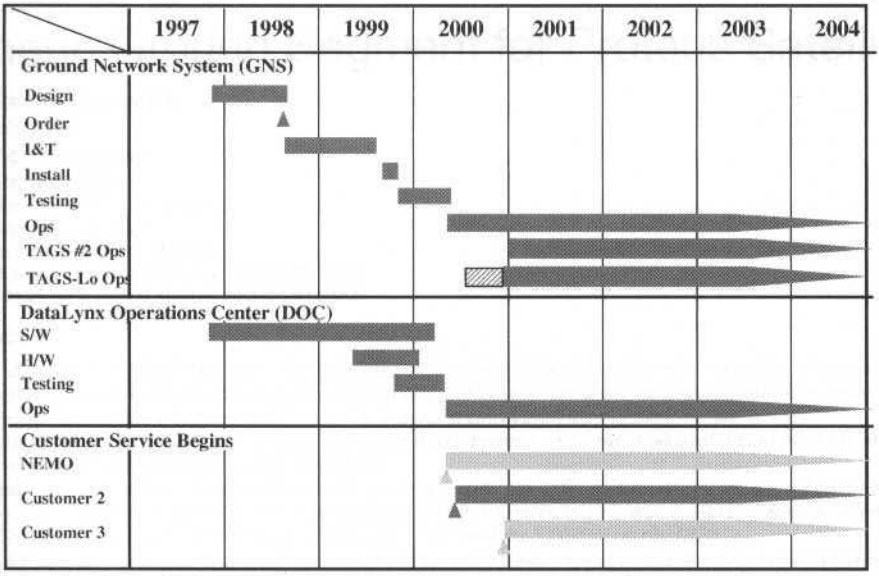


Fig. 6 DataLynx Development Schedule

Conclusion

ATSC has identified a business opportunity providing ground services to the rapidly expanding commercial space market, as well as to the traditional government markets. This opportunity capitalizes upon the growing number of ventures that need ground services, but which individually cannot fully utilize the capacity of dedicated facilities. ATSC proposes to meet the needs of many such customers by building DataLynx™, a network of facilities located in key geographical locations, offering utility-like services from those facilities. Whether the customer's space application is remote sensing, Earth and space science, or telecommunications, DataLynx™ will provide a broad and flexible solution for their satellite control and management requirements. DataLynx™ will allow the customer to focus their critical resources on their core business with the assurance that their space assets are being managed by the world's leaders in spacecraft operations. ATSC has been involved in more than 700 space missions since Explorer I in 1958.

DataLynx™ is being designed to provide an economical, flexible and expandable system using both proven technologies for reliability, and advanced technologies, such as automation, for efficient and cost-effective operations. Our approach takes advantage of ATSC's more than 45 years of experience in designing, building, operating, maintaining, and upgrading ground tracking stations and control centers throughout the world.

- An onboard GPS which reduces the cost of the operations and the cost of the ground location system and enables dating of TM and TC packets in universal time.

- The system is overall sized on the nominal operating case and not on exceptional cases. Thus, all onboard anomalies which may call the integrity of the satellite into question cause change to survival mode, the reaction times required being such that they do not need hot backups onboard or on the ground.

PGGS External Interfaces

Figure 1 shows the PGGS external interfaces, located between the Proteus satellite and the Mission Center specific to a user application.

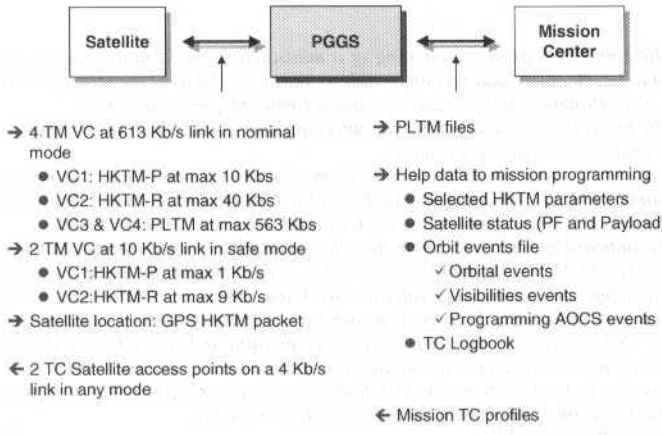


Fig. 1 PGGS external interfaces

Satellite - PGGS IF

The onboard/ground interface is fully compatible with the CCSDS packet TM/TC recommendations, which has the advantage of being a standard protocol. The software enabling its implementation is available and validated. The use of detector codes and error correctors (Reed-Solomon for Proteus) and the TC automatic retransmission protocol COP1 significantly improves the performance and quality of the TM/TC links.

In satellite nominal operating mode, the TM link is split into 4 virtual channels (VC). 2 are allocated to housekeeping TM (HKTM), the first one concerns the TM sent by the satellite during passes when ground station is visible (HKTM-P), the other corresponds to the housekeeping TM stored in memory (HKTM-R). The 2 latter channels are proposed to the mission and enable dump of stored payload TM into the mass memory. The HKTM-R and PLTM telemetry dumps are performed on Command - Control Center programming.

In survival mode, only the HKTM channels are used with reduced digital rates.

The single-band TC channel enables transmission of TCs for the platform and for the payload on 2 hot standby onboard decoders.

PGGS – Mission center IF

The PGGS supplies the payload telemetry and data to the Mission Center enabling it to elaborate future payload programming:

- Satellite surveillance parameters and a current satellite status resulting from last HKTM processing.
- Orbit events file giving, for the 10 days to come, all satellite events related to the orbit. For instance, dates when specific orbit points are passed, earth terminal visibility dates or orbit and altitude correction maneuver programming dates.

RBCM - J. of the Braz. Soc. Mechanical Sciences ISSN 0100-7386
Vol. XXI – Special Issue - 1999 - pp. 10-16 Printed in Brazil

Generic Ground Segment for Proteus Satellites

Jean Michel Tourraille
Guy Laborde
Frederic Lemagner
Jacques Mongis
Cnes, Centre Spatial de Toulouse
18, avenue Edouard Belin
31401 Toulouse Cedex 4, France
jean-michel.tourraille@cnes.fr guy.laborde@cnes.fr
frederic.lemagner@cnes.fr jacques.mongis@cnes.fr

Abstract

CNES is developing the PROTEUS project consisting of a minisatellite platform and an associated ground segment. The system is designed to be adaptable and reconfigurable so that it can be used for scientific missions or applications. A mission uses one or more Proteus satellites and a ground segment adapted to its needs.
The targets fixed for the project are reduction of initial developpment costs, rapid adaptability to platform and ground segment mission needs and simple system operability.
The ground segment, called "Proteus Generic Ground Segment" (PGGS) consists of one or more Telecommand and Telemetry Earth Terminal (TTCET), a Command and Control Center (CCC) and a Data Communication Network (DCN). The Mission Centers (MC) are connected to the CCC and the TTCETs via the DCN.
The TTCET sets up the onboard/ground link, sends the telecommands received from the CCC and transmits satellite telemetry to the CCC and to the MCs. Completely automated, it doesn't require the presence of operators.
The CCC ensures the telemetry processing, satellite orbit and attitude control functions, the generation and transmission of platform telecommands, the reception and transmission of mission telecommands from the MC. The operating modes of the CCC depend on the needs of the user mission and range from partially automatic operation during working hours and on working days to all manual operation 24 hours-a-day.
The aim of the presentation is to show how the PGGS attains the generic product development cost reduction and mission adaptation targets together with the operating cost reduction targets.
***Keywords**: CNES, Proteus, PGGS, CCSDS*

Introduction

At the start of the 1990's, CNES initialized the PROTEUS (Reconfigurable Platform for Observation, Telecommunications and Scientific Purposes) project the aim of which is the design of a reconfigurable generic system capable of adapting to various earth observation, telecommunications and basic research missions.

The system consists of 2 entities:

- A minisatellite platform, adaptable to most commercial launchers, with a mass at lift-off of 500 kg including a payload of 250 kg. Its flight envelope is between 450 and 1500 km on circular orbits with any inclination. The platform can deliver to the payload up to 250 W continuously. The 3-axis controlled attitude can ensure earth-pointed or inertial missions.
- A ground segment called "Proteus Generic Ground Segment" (PGGS), consisting of a Command - Control Center and one or more ground earth terminals connected by a data transmission network.

The platform and the PGGS are developed in parallel with the same requirements:

- Reduction in initial development costs and rapid adaptation at least cost to specific mission needs in order to facilitate carrying out of space programs. These requirements lead to maximum adoption of standards, especially for communications, favored use of "off-the-shelf" products, to the detriment of specific developments, and a highly modular architecture.
- Simple operability to reduce platform operating costs. The basic axiom is that the operations of a Proteus satellite are achieved from a control center and a single earth terminal which implies high satellite autonomy. For this, Proteus has:
  - A mass memory enabling storage of payload telemetry (PLTM) and satellite housekeeping telemetry (HKTM).
  - A memory enabling storage of platform telecommands (TC-PF) and payload telecommands (TC-PL) with deferred execution. These memories are sized to guarantee 7 days of satellite autonomy in nominal mode.

- A report of platform and payload TC transmissions to satellite (TC logbook).

The PGGS receives a payload program from the Mission Center in the form of an ordered sequence of TCs ready for transmission to the satellite. Transferring responsibility for generation of payload programming TCs to the Mission Center makes the PGGS very generic and independent of the mission.

PGGS functions, operational concepts and architecture

The dedicated PGGS functions were defined from the study of the generic Proteus system. The general Proteus requirements led to the definition of the operational concepts. The PGGS architecture stems from the functions to be conducted and the operational concepts laid down.

PGGS functions

For a given mission, the PGGS is a part of the mission ground segment. It does not ensure all the functions of the mission but all those required for final orbit acquisition and station keeping of the satellite or satellites of the mission.

PGGS functions:

- Satellite surveillance and technical control
  - These functions consist in checking, thanks to housekeeping telemetry HKTM processing, that the status of the satellite is satisfactory for mission needs and for transmitting telecommands to maintain normal satellite operation.
- Orbit and attitude control
  - Satellite orbit determination is performed by the PGGS from the GPS data received in the HKTM. If an orbit correction is required, the PGGS generates the control commands which are sent by TC and executed by the satellite. Attitude control is performed automatically onboard from the GPS data and the sensor information. The PGGS periodically updates the attitude control onboard model parameters.
- Payload service
  - This consists in transmitting the received payload telemetry PLTM to the Mission Center, checking the status of the payload thanks to HKTM processing and in performing the programming operations at the frequency dependent on mission requirements and satellite storage capacity.
- Satellite expert appraisal
  - Consists in conducting investigations in case of satellite malfunction or reports on its behavior. These operations are conducted by the operators of the Control Center or by external experts.

Operational concepts

The operational concepts stem from general requirements such as the need to execute imposed operations by the platform during working hours, compatibility with multiform mission ground segment organizations and compatibility with highly varied mission programming requirements.

The following concepts have been retained:

- The TTCET operates without operator and is controlled by the CCC.
- The CCC operates with an operator during working hours whose presence is required only for transmitting telecommands.
- The surveillance of the onboard and ground elements can be achieved automatically from the CCC; an anomaly must induce call for operator intervention.
- Satellite command generation is distributed between and attributed to groups with independent responsibility: the platform command generator, the AOCS command generator and the mission command generator.
- Satellite expert appraisal is decentralized from the CCC and can be conducted directly by the expert from his workstation.
- Satellite status surveillance is performed off-line at a frequency dependent on mission requirements.

PGGS architecture and operating modes

The PGGS is an assembly of 3 basic components: the Command - Control Center (CCC), the Data Communication Network (DCN) and the ground Telecommand – Telemetry Earth Terminal (TTCET). For the specific needs of a mission, several TTCETs can be used and the PGGS can interface with several Mission Centers.

Telemetry processing:

- The TTCET receives the surveillance satellite telemetry HKTM and the payload telemetry PLTM.
- Real-time telemetry HKTM-P is transmitted on reception by the TTCET to the CCC during the satellite passes. This enables the CCC to control transmission of TCs in relation to current satellite configuration.
- The HKTM-R telemetry stored onboard is dumped during the passes, stored in the TTCET and made available to the CCC which then has it at its disposal after the passes. This enables the CCC to perform off-line, the surveillance, the satellite monitoring and the orbit and attitude control. It is archived at the CCC throughout the life of the satellite.
- Payload telemetry PLTM stored onboard is dumped during the passes, stored in the TTCET and made available to the Mission Centers which then have it at their disposal after the passes.

Telecommand processing:

- The CCC generates the platform TCs and the orbit and attitude control TCs.
- The main Mission Center generates the payload programming TCs and transmits them to the CCC.
- The CCC sends all the TCs to the TTCET which, on request from the CCC, sets up the onboard/ground link, sends the TCs to the satellite and transmits, in real time, the TC acknowledgements received from the satellite at the CCC.

TTCET management processing:

- The CCC commands the TTCET by sending Remote Commands (RC). Station operating status is known via its Remote Monitoring (RM) telemetry sent to the CCC and to the mission centers.
- The antenna pointing data enabling monitoring of passes are generated by the CCC and sent to the station in the form of point sequences (date, elevation, azimuth, doppler).

Processing of surveillance and satellite expert appraisal data:

- The CCC archives and places, for the life of the satellite, the data required for the expert appraisal function (HKTM-R, logbook and command report, orbitography data) at the disposal of the internal CCC or external customers.
- The expert appraisal data can be consulted via the Data Remote Processing PC (DRPPC). The processing operations on these data are performed locally by the customer.

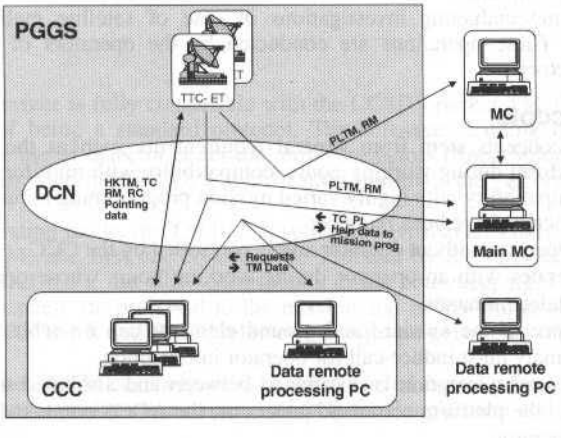


Fig. 2 PGGS architecture

TTCET - Proteus Telecommand – Telemetry Earth terminal

The TTCET consists of 3 subsystems:

- The "Radio Frequency" subsystem (SERF) sets up the onboard/ground link on order from the CCC. It ensures the reception of the signals delivered by the satellite (TM CCDS frames on 4 virtual circuits) and the transmittal of signals to the satellite (TC CCSDS frames received from the CCC). Reception is made in circular polarization diversity mode and transmission according to a polarization selected by the CCC.

- The "Base Band" subsystem (SEBB) interfaces with the SERF and the CCC and the Mission Centers. It receives the TCs and the pointing data from the CCC and transfers them to the SERF. It processes the TM frames received from the SERF by separation of virtual circuits with local storage of HKTM-R and PLTM. HKTM-P and the TC onboard reception acknowledges are sent in real time to the CCC.
- The "Time Frequency" subsystem distributes a time reference and a frequency reference to the SERF and to the SEBB.

Certain characteristics of the TTCET enable the development cost, recurrent cost and operational cost targets to be reached:

- The TTCET requires no operator interventions, the TCs are received from the CCC and directly sent to the satellite and TM is processed automatically. TTCET control is simple and performed via macrocommands sent by the CCC. The TTCET gives information concerning its status via the RM.
- The Proteus link budget is compatible with the use of a large-lobe ground antenna (3°). The TTCET therefore has no automatic tracking system, pass monitoring is achieved from the pointing data delivered by the CCC.
- The low-diameter antenna (3.10 m) is placed under a radome which greatly reduces the mechanical cost of the antenna. This radome also enables a reduction in the costs of the infrastructure by housing all the TTCET equipment.
- Maximum use of "off-the-shelf" equipment limits special developments.

DCN - Data transmission network

The interface specifications of the PGGS component data processing systems are all based on IP (Internal Protocol) transfer enabling real time transfer via IP service sockets and transfer of files by using the FTP (File Transfer Protocol) service. The Mission Center(s) must comply with the same interface specifications. The various local networks of the PGGS components are connected to the long distance IP network retained for the mission by standard routers.

The protocols used for transfer of PGGS data are:

- TCP-IP for real-time exchanges between the CCC and the TTCET (HKTM-P, TC, RC, RM) for transmitting TTCET RMs to the Mission Center(s).
- FTP for transfer of files (HKTM-R, PLTM, Pointing data, TC_PL, mission generation help data).
- HTTP and e-mail for data exchanged between the CCC and the expert DRPPCs.

Within the scope of the Proteus project requirements, the advantages in using an IP network are multiple:

- It is a standardized protocol supported both by the local communication networks and the long-distance communication network set up. This enables us to ensure in a unified manner the local and remote exchanges and favors the interoperability of the DCN with the private networks of the user missions.
- It enables factory products without special developments to be used for its implementation.
- This protocol offers operating security by controlling transmission and guaranteeing the integrity of the data exchanged.
- It is a mature protocol with an assured life thus preserving the long-term generic PGGS investments. As not incompatible with high flow rate technologies and meeting multimedia services, it will enable the DCN to easily follow technical communication developments which will be required for future missions.

CCC - Proteus command/control center

The development cost, recurrent cost and operational cost reduction objectives are ensured for the CCC by the design and manufacturing choices:

A highly modular architecture where the CCC is broken down into groups ensuring a logical set of functions. The inter-group communications are made via an IP network by favoring ASCII file interfaces by using the FTP protocol. This modularity enables the development of the subsystems to be conducted in parallel with an easy integration phase for the assembly. The CCC consists of 3 main subsystems:

- "Onboard/ground interface" subsystem (G1) which ensures the following functions:
  - Reception and processing of the real-time monitoring TM HKTM-P
  - Generation of platform TCs

- Reception of orbit and attitude control TCs and payload programming TCs
- Transmission of TCs to satellite and management of TC acknowledge protocol CCSDS (COP1)
- TTCET command and control

- "Orbit and Attitude" subsystem (G2) which ensures the following functions:
  - Orbit determination from GPS data and determination of orbit corrections to be programmed.
  - Generation of orbit TCs, TCs for correcting parameters of attitude control onboard models, antenna pointing data and orbit events file.
- "Satellite surveillance and archiving" subsystem which ensures the following functions:
  - Recovery and processing (demultiplexing, surveillance and archiving in a databank) of HKTM-R stored in TTCET. Supply of GPS data to G2.
  - Recovery of data to be archived produced by the various subsystems to be archived (logbook and command report, orbitography data)

The consultation function for the archived data at G3 and distribution of results is performed by a data server WWW. A customer can thus consult the archived TM parameters, the raw TM packets, the status of the satellite, the logbook and order reports and the operational documentation. The results of consultation requests are sent to the customer in the form of files or HTML pages. The customer uses either a standard workstation equipped with a navigator, or a specific station (DRPPC) equipped with a navigator and packages supplied by the CCC (synoptic animator for playing TM for instance). This technology offers many advantages for achieving the Proteus CCC targets:

- This technology is simple to employ by use of servers and navigators available on the market.
- It is fully compatible with the local and long-distance network specification used in the PGGS.
- It falls well within the modularity and adaptability framework required for the CCC. The number of customers depends on the satellite life phase (final orbit acquisition, in-flight and routine commissioning) and the needs specific to users missions.
- The expert appraisal stations are standardized and the server customers can be the CCC operators, the satellite specialists who follow up the operations inside or outside the CCC, and the experts monitoring the equipment.

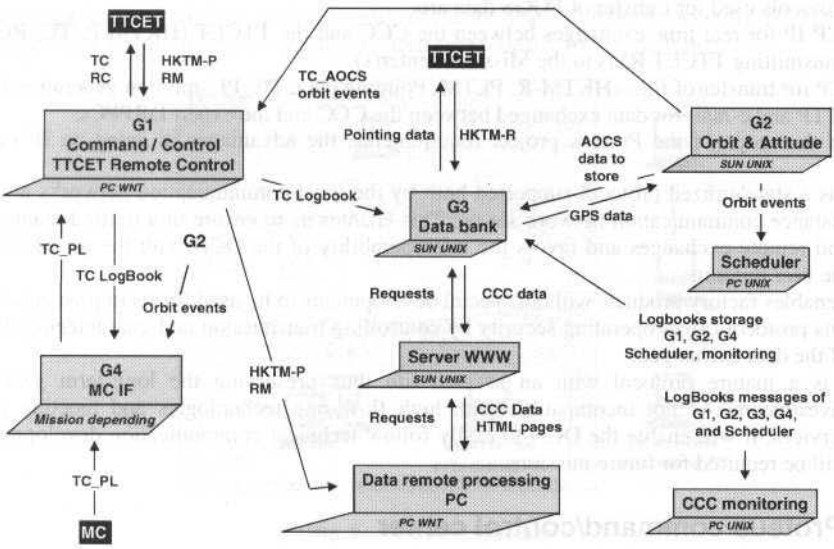


Fig. 3 CCC architecture and interfaces

In order to reduce operating costs and to facilitate the adaptability of the CCC to the operational requirements of the missions (from all manual with operators 24 hours-a-day to partially automatic with operator required only to send TCs), 2 subsystems were added to the CCC:

- "Scheduler" subsystem enabling control of CCC by activating tasks on various subsystems of the CCC and check for correct execution.

- "Warning surveillance" subsystem for surveillance of satellite and CCC warnings via logbooks and signaling of warnings by remote operator call.

These 2 subsystems are already operational in other control centers at CNES and are reused without adaptation for CCC needs.

The previously named subsystems (G1, G2, G3, WWW server, DRPPC, Scheduler and Surveillance) form the generic CCC which offers standard interfaces for the Mission Center requirements (see §2). For specific requirements, it is planned to add a specific group (G4) on which the processing operations required by a mission not covered by the generic system will be performed.

Conclusion

Initialized in 1997, the PGGS will be validated by the start of 1999. The first mission using a Proteus platform and the PGGS ground segment is an oceanic altimetric mission, Jason, fruit of cooperation between CNES, NASA and NOAA, launching of which is scheduled for the 1st quarter of the year 2000. In spite of a complex mission ground segment, the generic PGGS can be integrated without major adaptations. The second and the third missions planned are astronomy missions, Corot, launching planned for 2002 and Picasso planned for 2003. The developments performed to meet the Jason mission requirements have shown that the PGGS perfectly meets the mission's ground requirements. An earth observation mission following on from the Spot system and using Proteus satellites is now under study at CNES.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 17-22

ISSN 0100-7386
Printed in Brazil

Global Commercial Space Network (GCSN) Architecture

James Michael Stevens
Lockheed Martin Space Operations Company
7375 Executive Place
Seabrook, Maryland 20706 USA
michael.stevens@lmco.com

Worldwide space program budget pressures are leading agencies to consider structural and institutional changes to satellite operations.All worldwide space agencies, private industry, and academia are seeking additional satellite program cost savings by reducing operational support expenses. Use of commercial investment is saving substantial resources and allowing the funds saved to be redirected toward the development of advanced technologies. Outsourcing to industry allows the reduced workforce to concentrate on developing technology, which enables future missions to collect more data at significantly reduced cost. Outsourcing in this manner offers customers a method to buy what industry has to offer and make minor adaptations on the payload side to accommodate the commercial operations support.

Lockheed Martin's response is to establish a global commercial space network of satellite tracking stations. Our vision is for Lockheed Martin to become a major enterprise providing global space operations tracking and data acquisition services. The Lockheed Martin strategic plan provides a significantly better customer focus, develops new operational support technologies, and encourages customers to use a commercial space-tracking network.

This paper describes the commercial data acquisition services being implemented by Lockheed Martin through the establishment of worldwide Tracking & Data Acquisition stations. In addition, this paper describes how Lockheed Martin is methodically implementing its Global Commercial Space Network (GCSN) through the use of strategic alliances with all the world's providers of network type services and Lockheed Martin investments where required.

Lockheed Martin Space Operations Company will also provide this type support to the CSOC Commercial Operations Team. We will assist them in implementing commercial network operations support to the NASA Space Operations Management Office (SOMO) by performing a CSOC non-NASA T&DA broker role on a commercial basis where possible.

The Lockheed Martin Space Operations Company (LM SOC) headquarters are in Houston, Texas. The overall LM SOC organization is shown in Fig. 1.

The Global Commercial Space Network (GCSN) is a part of the East Coast Operation (ECO) located in Seabrook, Maryland. The overall organization of the ECO is shown in Fig. 2.

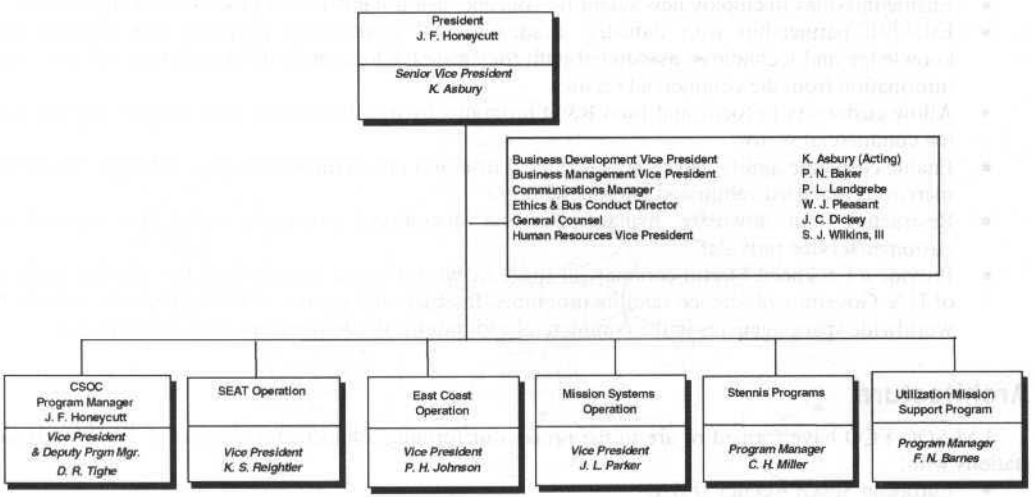


Fig. 1 Lockheed Martin Space Operations Company

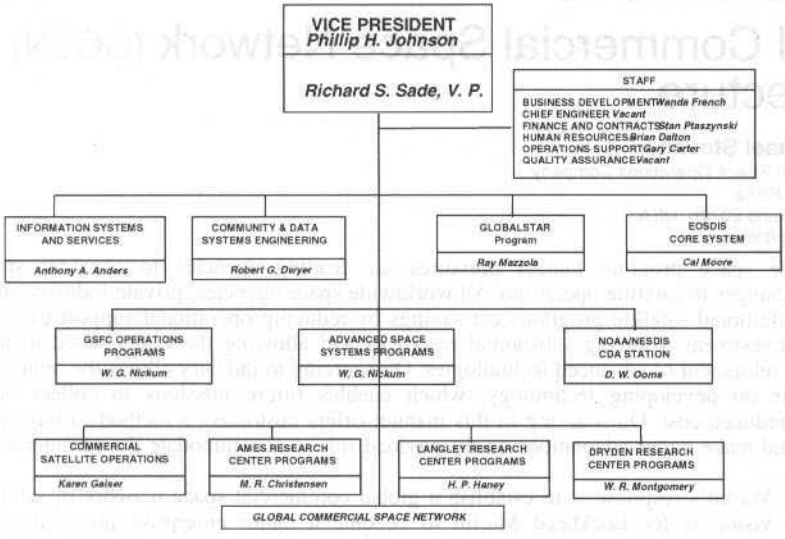


Fig. 2 East Coast Operation

Global Commercial Space Network Program Objectives

- Develop and maintain a cost effective low earth orbit (LEO) satellite network system of worldwide satellite data acquisition stations with a central Network Control Center (NCC). This network will acquire and process the information needed to support customers' scientific and technological objectives.
- Assist customers in defining their mission network support requirements and developing T&DA partnerships.
- Provide the essence of commercial practice, i.e., deliver support in accordance with performance specifications rather than design and concepts provided by customers. This approach allows Lockheed Martin GCSN to use risk management strategies that are not always available to customers.
- Enable missions to employ new scientific concepts using standardized procedures and practices.
- Establish partnership with industry, academia, and government agencies that expands the knowledge and technology associated with the space tracking and data acquisition services, and information from the commercial sector.
- Allow customers to focus and fund R&D programs by transferring network support functions to the commercial sector.
- Enable corporate agility and prompt technical insertion into commercial space network operations increasing standardization and interoperability.
- Restructure and downsize management and operational processes using the concept of customer/service provider.
- Provide a Lockheed Martin commercial space network venture serving the LEO satellite markets of U.S. Government science satellite programs; International science satellite programs; and other worldwide space agencies in the commercial and industrial satellite markets at reduced cost.

Architecture

LM SOC ECO have formed or are in the process of forming alliances for the use of T&DA ground stations with:

- European Space Agency (ESA)
- Ingenieriay Servicios AeroEspaciales, Spain (INSA)
- Russian Space Agency (RSA)
- Ukrainian Space Agency (NSAW)

- Norwegian Space Agency (NSC)
- French National Space Agency (CNES)
- CSIR, South Africa (HBK)
- German Space Agency (DLR)
- TELSTRA, Australia (PITC)
- Italian Space Agency (ASI)
- EOSAT Corp., Oklahoma
- Universal Space Network, Pennsylvania (USN)
- Mitsubishi Corporation
- Kongsberg Spacetec, Norway (KLMSPS)
- DASA
- University of Chile, Santiago, Chile (AGO)
- TELESPAZIO
- Indian Space Research Organization (ISRO)
- Swedish Space Agency

Lockheed Martin GCSN Architecture

Strategic alliances have thus far resulted in the possibilities of partnerships with worldwide T&DA providers (Fig. 3).

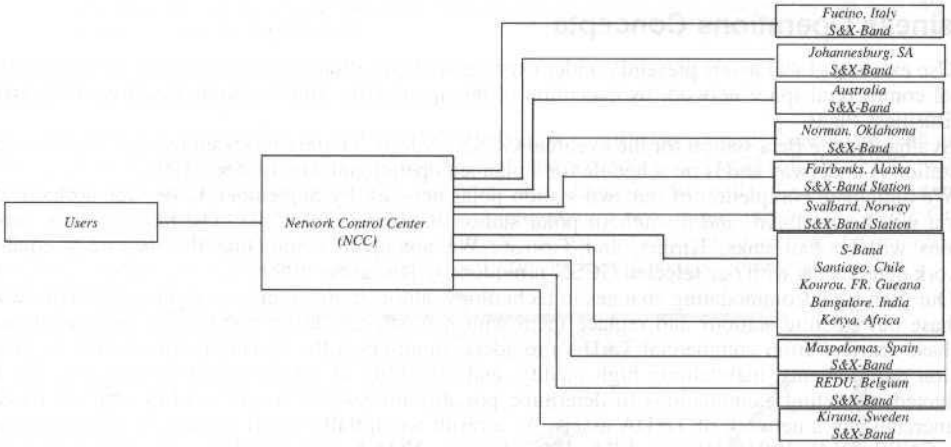


Fig. 3 Lockheed Martin GCSN Architecture

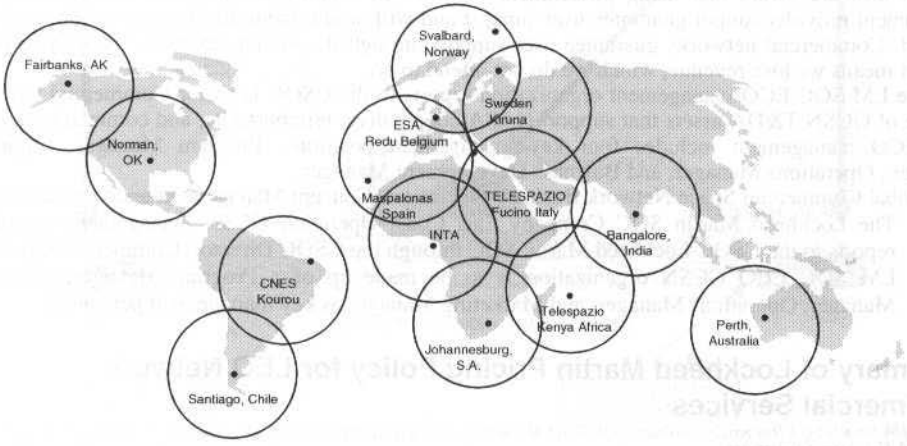


Fig. 4 GCSN Data Acquisition Station Locations Architecture

Table 1 Typical Site Characteristics

Technical Characteristics	S-Band Capabilities	X-Band
Antenna		
Mount	3 Axis, 7° Tilt	3 Axis, 7° Tilt
Tracking Modes	Auto, Program, Manual	Auto, Program, Manual
Tacking Rates	.002 °/sec to 10 °/sec	.002 °/sec to 10 °/sec
Slew Rate	7.5 °/sec E1, 15 °/sec Az	7.5 °/sec E1, 15 °/sec Az
Acceleration	10 °/sec	10 °/sec
EIRP	94 dBm at 5°	94 dBm at 5°
Min Gain/Noise Temp	G/T >= 22.6 dBk at 5°	G/T >= 35.4 dBk at 5°
Uplink Parameters		
Modulation	PCM/BPSK, PSK, FSK, PM PM Direct Spread Spectrum	Future Capability
Data Rates	1 bps to 96 Kbps	
Subcarrier Frequency	100 to 16,000 Hz	
Symbol Rate Ratio	8:1 to 1024:1	
Downlink Parameters		
Modulation	PCM/FM, PM, BPSK, QPSK SQPSK, USQPSK, UAQPSK	BPSK, QPSK, SQPSK, USQPSK, UAQPSK
Data Rates	UP to 10 Mbits/sec	Up to 150 Mbits/sec
Data Error Correction	Reed-Solomon, CRC Viterbi	Reed-Solomon, CRC, Viterbi

Business Operations Concepts

Use existing global assets presently under existing strategic alliance to complete the development of a global commercial space network by execution of the appropriate MOU's, Joint Ventures, Partnerships, and Business Plans.

Svalbard is the Beta station for the eventual GCSN. A Joint Venture has been formed, construction of the station is underway and is on schedule for a planned operational date of April 1999.

We anticipate completion of our two-station polar network by September 1999. Our northern polar station will be Svalbard, and a southern polar station is planned for Chile. The backup polar network stations will be Fairbanks, Kiruna, and Tromso. We anticipate completing the necessary equatorial network agreements with our selected GCSN providers by December 1999.

Our plan for accommodating changes in technology, automation, communications, and frequencies is to phase out existing stations and replace them with new stations using commercial investment and/or purchase services from commercial T&DA providers. In all cases the transition criteria will be meeting mission requirements, maintaining high quality, and reliability of service at the lowest cost. We have conducted marketing examinations to determine possible interest for use of a LEO network based on commercializing a network of T&DA assets. As a result we initially found at least 36 pending missions from CNES, DLR, ISRO, Boeing, ESA, ISIS, Taiwan, NOAA, and NASDA with various degrees of interest in using such a network.

We also are aware that using government facilities to support reimbursable missions means the government provider cannot guarantee user support and will not be liable for the successful provision of support. Commercial networks guarantee user support through its pricing structure. Inability to provide support means we lose revenue, which we do not intend to do.

The LM SOC ECO management organization reports to the CSOC Director Commercial Operations for use of GCSN T&DA assets that support NASA cooperative, reimbursable, and commercial missions. The ECO management includes four key management positions; Program Manager, Engineering Manager, Operations Manager, and Business Development Manager.

Global Commercial Space Network Business Operations Concept Management and Organization:

- The Lockheed Martin SOC Company East Coast Operation GCSN management organization reports to the CSOC Lockheed Martin SOC through the CSOC Director, Commercial Operations
- LM SOC ECO GCSN organization is to be made up of a Program Manager, Engineering Manager, Operations Manager, and Marketing Manager as key management personnel

Summary of Lockheed Martin Pricing Policy for LEO Network Commercial Services

LM Pricing Summary

- Proposed satellite T&DA services are separated into three major categories:

- Operational/routine pass support for both S-band and X/S-band T&DA support, services are priced on a $ per pass basis
- Launch & Early Pass (LEOP) support, services are priced on a $ per pass basis/for early orbit and a $ per hour for prelaunch and launch
- Contingency/emergency support is expected to be no more than an additional 40-50% of operational pass price

LM Pricing Assumptions

- All prices are based on recovery of operations costs and the precise impact of savings will not be known until detailed T&DA M&O cost data is made available and incorporated into each business plan. Other pricing assumptions include:
  - T&DA service pricing is based on LEO mission requirements only.
  - "Pass" is defined as any use of a ground network antenna asset. For routine service, a length of pass is assumed from 3-20 minutes (the satellite duration over the ground antenna). For launch & EO support, a pass is defined as anytime an antenna asset is "tied-up" in support of mission requirements (e.g., for simulation, training & test as well as actual launch)
- Services not included in the wrapped pass price:
  - Intent is to define a "standard" pass cost for routine, launch, and EO operations, with variable charges for high-rate communications and pre-launch engineering support defined by user/mission needs as unique additive requirements
  - Pass price for routine services includes all services to provide end-to-end TT&C network capability.
  - Per pass cost also includes all associated business support costs and delivery of low rate (512 Kbps) data to the GCSN NCC.

Operations Concept

The GCSN NCC will be operational April 1999. The NCC will operationally coordinate with all the GCSN providers. Day-to-day satellite scheduling of the GCSN and O&M of the NCC will be accomplished by Lockheed Martin through the NCC. Lockheed Martin will operate as the point-of-contact/broker and mission planning element for implementing a new requirement that use the GCSN. LM SOC ECO will participate as necessary as a GCSN advisor and representative to CSOC commercial operations by supporting CSOC mission planning activities including panels, committees, and etc.

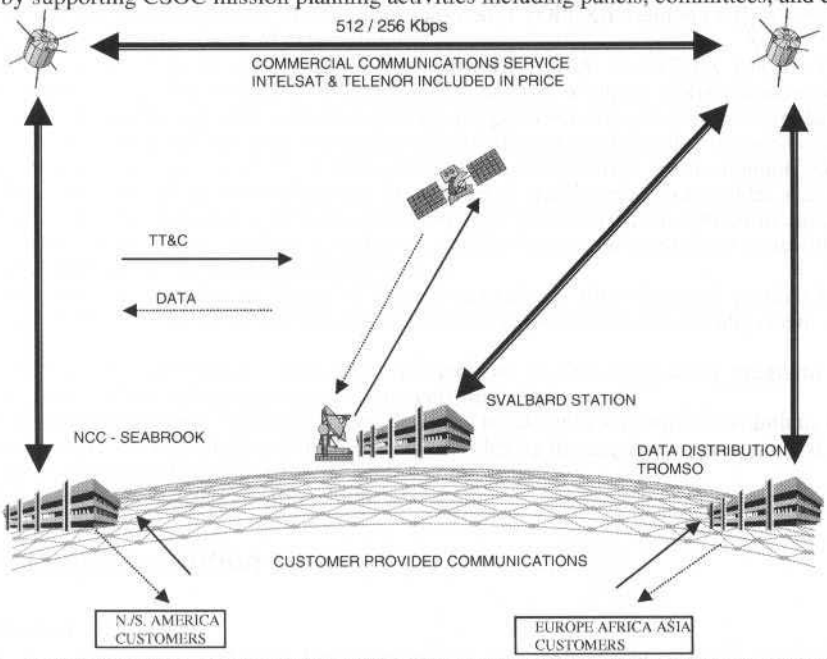


Fig. 5 System Overview - Representative Station of the LM GCSNOperations Concept Lockheed Martin Responsibilities:

- Commercial, cooperative, and reimbursable requirements and specifications to be provided to GCSN by LM SOC. LM SOC ECO coordinates within the GCSN providers.
- Installation and integration of the NCC - underway - operational April 1999
- Day to day spacecraft scheduling, point-of-contact/broker for remote operations of the stations, and O&M of the NCC.
- Sustaining engineering and Mission Planning for implementing new mission requirements.
- Marketing services to be provided by LM SOC ECO for commercial support above and beyond CSOC.

LM SOC ECO will participate as necessary as a GCSN advisor and representative to LM SOC Commercial Operations by supporting all mission-planning activities including panels, committees, etc.

We want to thank you for inviting us to participate in SCD-II and allowing us to present our GCSN objectives. We believe that by early 2000 LM SOC will have an extremely efficient global commercial network supporting space activities.

We stand ready to discuss using the Lockheed Martin GCSN services to provide the support you need and want at cost effective rates through our very generous pricing plan.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 23-28
ISSN 0100-7386
Printed in Brazil

Jason Ground System Architecture and Operation Concept

Gérard Zaouche
Centre National d'Etudes Spatiales (CNES)
18 Avenue Edouard Belin, 31055 Toulouse Cedex, France
gerard.zaouche@cnes.fr

Abstract

The Jason-1 satellite is the first from the set of satellites which will constitute the Jason Altimetry Program. The importance of altimetry data to better understand the ocean circulation and its impact on the climate of the Earth led to the TOPEX/Poseidon mission. Thus it is the aim of the follow-on to TOPEX/Poseidon, the Jason missions, to provide on a continuous basis for a minimal period of 20 years or more, the same high accuracy altimetric measurements designed to study ocean circulation and sea surface elevation.

The Jason Satellite includes the launcher adapter, the satellite bus (derived from the PROTEUS platform currently under in development at CNES) and the instruments constituting the payload (a two frequency altimeter called Poseidon2, a three frequency radiometer JMR, the DORIS on board package, a laser retroreflector array, a GPS space receiver called TRSR).

The Jason Ground System (JGS) consists of a satellite control ground system and a mission ground system. The main feature of the Jason Ground System is to provide two Satellite Control Centers with one located in Toulouse (France) active during the first phases of the satellite's life, and the other located in Pasadena (USA) active during the routine phase. Each location includes a Mission Center which will be active during the life of the satellite.

This paper begins with an overview of the Jason System and its mission. It then details the Jason Ground System, its functions and its architecture distributed over the two sides of the Atlantic ocean. It finally describes the planned phases in the life of the Ground System (according to the satellite life phases) and the associated operations (transition between control centers, routine phase operation, transition between operational teams, ...).

Keywords: *Jason, Altimetry, Ground System, Command-Control Ground System, Mission Ground System, Architecture, Validation, Operations.*

Introduction

The Jason satellite is the successor of the highly successful TOPEX/Poseidon (T/P) spacecraft which has achieved science objectives beyond expectations.

The mission is a follow-on cooperative effort between NASA and CNES for the continuous observation of the oceans. The objective is to obtain a long-term accurate global measurement of the ocean topography in order to improve our understanding and previsions of global ocean circulation.

Like T/P, Jason will provide accurate sea surface topography data to determine the general circulation of the oceans and evaluate its role on the Earth's climate. Other objectives are to monitor and interpret regional and global sea level change, to improve knowledge of (long period) ocean tides and to observe and use wave-height and wind speed for marine meteorology. Additional contribution of Jason data are foreseen in various domains like geodesy, geophysics, shallow water and coastal environment, enclosed sea circulation, inland water and land topography survey.

In addition, it is an objective of Jason to provide a near-real time data and product service for operational activities such as marine nowcasting and numerical prediction of sea state, ocean circulation and weather.

Use of T/P data in combination with other remote and in-situ observation programs is highly encouraged as a means for evaluating potential Jason performances.

Each component of the Jason system has been designed to take into account the evolutions in satellite bus and instrument technology, and the improvements provided by the acquired experience and the highly demanding performance requirements of the scientific community.

Launch of Jason-1 is expected in spring 2000.

Jason System Description

Jason Mission

The importance of altimetry data to better understand the ocean circulation and its impact on the climate of the Earth led to the T/P mission. With the launch of this mission began a data collection that

must continue into the next century in order to monitor the inter-annual evolution of the ocean dynamical signals and separate transient phenomena from secular variations.

To this end, the Jason Project will design, develop, integrate and test a satellite system with its suite of instruments to be launched in spring 2000 to ensure data continuity between T/P and Jason.

Jason will use an Earth orbiting satellite equipped with a radar altimeter and other instruments to directly measure sea surface elevation along a fixed grid of sub-satellite ground tracks and thereby continue the data collection started with T/P. Jason shall use the same ground track pattern as T/P.

Jason measures the distance from the satellite to the sea surface to within approximately two and a half centimeters (1s measurement accuracy).

Precise orbit determination allows to locate the spacecraft to within two or three centimeters and combination with radar measurement allows Scientists to achieve the Jason primary mission objective -- the production of accurate topographic maps of the world ocean.

The satellite orbits at an altitude of 1336 kilometers above the Earth with a 66 degrees inclination. In approximately 10 days Jason completes 127 orbits, namely a cycle and records sea-level measurements for the entire globe.

Jason Satellite

The satellite (Fig. 1) includes the launcher adapter, the satellite bus and the instruments constituting the payload. The satellite bus is comprised of a platform and a payload module.

The platform comprises the support functions for on orbit operations, including provision of electrical power, command and data handling, telecommunications, thermal control, propulsion/orbit maintenance, and primary structure.

The payload module provides mechanical, electrical, thermal and dynamical support to the Jason instruments.

The spacecraft will be the first satellite to use the new small platform called PROTEUS, built in partnership between CNES and AEROSPATIALE.

The technology improvement, electronics compact packaging, led to a tremendous decrease of the satellite resource needs. As an example, the overall mass of the JASON satellite will be less than 500 kg as compared to 2400 kg for the T/P satellite.

Jason Satellite Performances:

Satellite mass:	500 kg
Payload module size:	954 x 954 x 1218
Satellite power:	370 W
Storage capacity:	2 Gbits (EOL)
Lifetime:	Up to 5 years
Downlink capacity:	650 kbps
Bus dry mass:	245 kg
Uplink capacity:	4 kbps
Bus consumption:	170 W
Pointing accuracy:	0.035°
Bus size (mm):	954 x 954 x 1000
ΔV capacity:	Up to 120 m/s

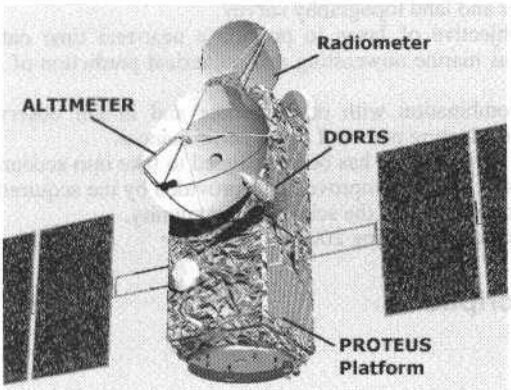


Fig. 1 Jason Satellite

The nominal Jason payload consists of:

- a two frequency altimeter called Poseidon-2 and its antenna to send radar pulses.
- a three frequency radiometer (JMR) and its antenna to estimate the atmospheric water vapor content in the nadir column, in order to correct the radar altimetry measurement.
- a DORIS receiver which is a radio tracking system developed by CNES, using the Doppler shift of ground beacons to provide an unprecedented precision orbit determination. It is comprised of a Doppler receiver, an Ultra Stable Oscillator and the associated antenna.
- a Turbo-Rogue Space Receiver which is a GPS receiver.
- a Laser Retroreflector to reflect incoming laser beams from the ground allowing the calibration of the radial position of the satellite.

The PROTEUS (Plate forme Reconfigurable pour l'Observation de la terre, les Telecommunications et les Utilisations Scientifiques) program intends to develop a standard platform corresponding to a satellite mass (including launcher adapter) around 500 Kg. Attitude control is based on star tracker, gyros and reaction wheels. The propulsion system uses an hydrazine module with a spherical tank to allow final orbit acquisition and maintenance with comfortable margin with respect to foreseen operational lifetime. The data handling is centralized, built around a 31750 microprocessor and uses a 1553 data bus for payload data interfaces. Solid state recorders provide the required data storage. An S band configuration is used for scientific and housekeeping telemetry and CCSDS protocol is applied in both forward and return link mode (use of Virtual Channels). Convolutional coding is also applied to telemetry.

Jason Launcher

Launch of the Jason satellite will be provided by NASA. The launch vehicle will be a Delta II 7920, a 2-stage liquid rocket with 9 solid propellant motors strapped to the first stage. Launch will be from Vandenberg Air Force Base over the Western Test Range. That launch is co-manifested with another NASA satellite named TIMED.

Jason Ground System (JGS)

The resources of the JGS are used to support mission planning, telemetry data capture and staging, satellite monitoring, command preparation, satellite and sensor control, satellite localization and orbit control, support system scheduling, housekeeping data and science data archiving, science data processing, data verification and distribution for the life of the mission.

The JGS is comprised of a satellite control center located in Toulouse, an operational control center located in Pasadena, and mission centers located in CNES- Toulouse and Pasadena. Three small ground stations located in Toulouse, Poker Flat and Wallops allow nearly one visibility opportunity during each orbit.

The entire JGS architecture is detailed in section 4.

Three different data products shall be produced and distributed to the users:

- Operational Science Data Record (OSDR)
  - The OSDR shall include wind, wave and altimetric information and shall be distributed in near real time (75% of the data taken shall be made available to the users within 3 hours from acquisition and 95% of the data within 5 hours)
- Interim Geophysical Data Record (IGDR)
  - An IGDR (for instance used for mesoscale and El Nino type event studies) shall be delivered within 3 to 5 days maximum. The IGDR shall be of the same format and content as the GDR, but the accuracy of some data items (mainly orbit) may not be fully checked.
- Geophysical Data Record (GDR)
  - The final Geophysical Data Record (GDR) shall include fully corrected altimeter data with ancillary geophysical information suitable for carrying out the scientific objectives of the mission and shall be delivered within 30 days.

In addition to these «standard» products, a certain number of specific products will be made available upon request by users for other applications. For example the SGDR (Sensor/Geophysical Data Record) product contains all information included in the GDR plus information from level 0 and level 1b (for instance, waveforms) altimeter data. It is generated for altimeter experts interested in qualifying the instrument performance and it also meets requirements from science users looking at altimeter measurements for unusual ocean conditions.

Jason Ground System Architecture

The Jason Ground System (JGS) consists of a control ground system, a mission ground system and a data communication network. Some JGS elements are permanent and used by the JGS during all mission phases, while others elements are additional and only used during the Launch phase.

Permanent elements:

- The control ground system includes:
  - a Satellite Control Center (JCCC) located in CNES-Toulouse.
    This center monitors the satellite during the whole mission life time. Satellite control and operations are executed from this center until the end of the assessment phase. Following this milestone, operations plan preparation, navigation functions, platform configuration changes as well as performance and trends analysis are still this center's responsibility while routine operations including satellite activities scheduling, command plan preparation, command transmission and telemetry acquisition and routing are transferred to the POCC.
  - a Project Operation Control Center located in JPL-Pasadena (POCC)
    This control center (JTCCS and JSEQ) will be operational from the end of the assessment phase and will control the satellite and the associated instruments for the remainder of the mission.
  - Earth Terminals
    The planned configuration is to have one JPL earth terminal located in Poker Flat, Alaska and a second CNES earth terminal in Aussaguel (France). A third JPL earth terminal is available at Wallops for backup. The Earth Terminals performs satellite telemetry capture, Virtual Channels splitting, recording and distribution to the control centers (housekeeping telemetry in real time and organized in files) and to the mission centers (payload telemetry organized in files). The earth terminals also perform the uplink commanding to the satellite.
    At any time, only one control center directly interfaces with the satellite and is called the -- active control center --.

The mission ground system includes:

- a CNES mission system (part of SSALTO: Segment Sol Multimission Altimétrie et Orbitographie) including:
  - a CNES mission center located near the CNES facility
    The mission center functions consist in instrument programming and monitoring (altimeter and Doris), commands requests generation (altimeter and Doris), mission management and operation plan definition, Precise Orbit Determination (POD), algorithm definition and POD data production and validation, scientific altimeter data processing and validation of altimetry product, product distribution and archiving.
  - the Doris system beacons network
- a NASA mission center (part of the JPL POCC)
  The mission center (JSDS) functions consist in instrument programming and monitoring (Radiometer and GPS), command requests generation (Radiometer and GPS), scientific altimeter data processing and validation of altimetry product in parallel with the CNES mission center, operational altimetry data processing, data distribution and archiving.
- The Data Communication Network (DCN) which allows all the JGS elements to be linked. It is partitioned into:
  - a CNES Network (DCNCNES) which links all the elements of the CNES part
  - a JPL Network (DCNJPL) which links all the elements of the JPL part
  - a Network which links the DCNCNES and DCNJPL

Additional elements:

- a CNES Earth Terminal named SBANDET-HBK which is a S-BAND Earth Terminal with a special kit installed to make the terminal compatible with the Jason satellite and ground system. SBANDET-HBK will be used during the launch phase only to cover the satellite-launcher separation and to add some visibility periods.
- a CNES Main Control Room (MCR) and a Spacecraft Specialist Room (SSR) from where Jason project managers and the Jason specialists will be able to follow the HK telemetry and to control the operations. These rooms are used during the launch phase only.
- one or more Earth Terminals which will be able to provide Angular Measurements to locate the satellite in case of a large dispersion of the launcher Delta 2 (ET from CNES or NASA 2GHz network)
- a CNES Orbitography Operational Center (OOC) located in CNES facility during the launch phase . It will be able to coordinate and provide 2 GHz Earth Terminal (listed above) with pointing data

External elements: These elements provide support to the JGS during the Jason life time.

- IERS, MEUDON, French MTO (French MeT Office), GPS network, Laser Database, Doris master beacons, PODAAC (Physical Oceanography Distributed Active Archive Center) – PODAAC will provide data archiving and data products distribution for the JPL elements.

The following diagram (Fig. 2) shows the high level interfaces between the JGS elements. A solid line indicates that the connected elements will interface throughout the mission, a dashed line indicates that the connected elements only interface during the beginning of the mission (additional elements).

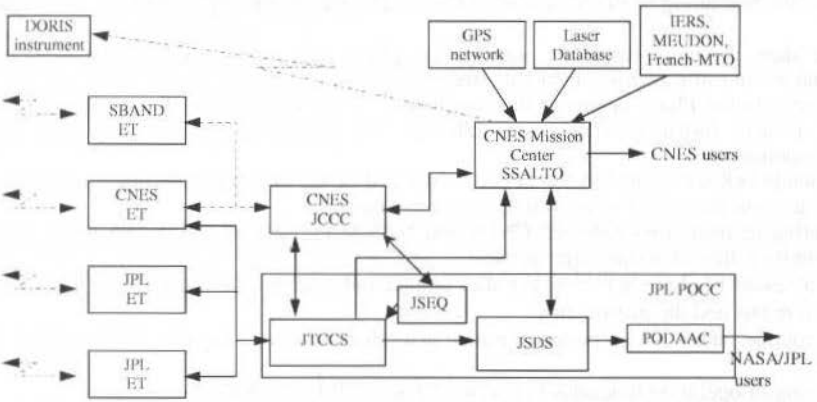


Fig. 2 JGS Architecture

JGS Operations

The Jason mission is conveniently broken into five phases of operation, as listed below (Fig. 3):

(1) Launch phase
(2) Assessment phase
(3) Verification Phase
(4) Observational Phase
(5) Extended Operational Phase

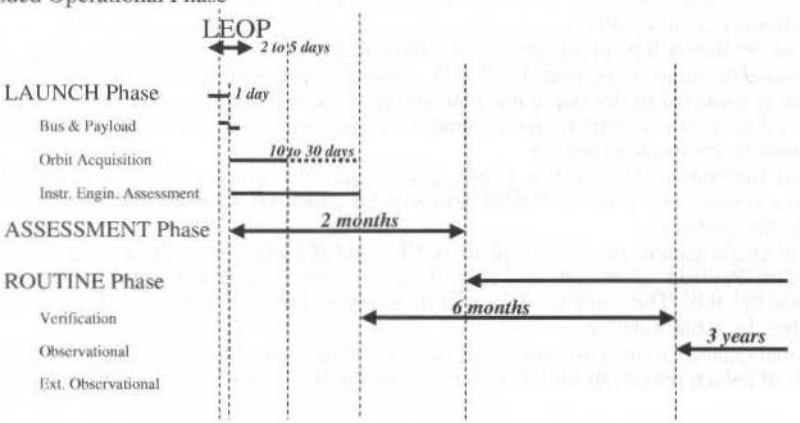


Fig. 3 Jason Mission Phases

The Launch Phase begins one day before launch and ends shortly after injection of the satellite into the injection orbit. Activities during this phase center on preparing the satellite for launch (integration with the launch vehicle, communication testing, and operation team testing and training) and launch operations.

The Assessment Phase begins at the end of the launch phase, and ends when all satellite and instruments systems have been deployed, activated, and functionally certified, the satellite is in the operational orbit and the ground system is ready to operate routinely. This phase includes a series of propulsive maneuvers designed to place the satellite onto its operational orbit. The satellite contractor and instrument suppliers are actively involved during this phase.

The duration of the assessment phase is 2 months maximum.

The Verification Phase overlaps the Assessment Phase in time, beginning when the satellite is in the operational orbit with the instruments engineering assessment completed, and continuing until the data

received from the satellite, the instruments and the processing algorithms are satisfactorily calibrated and validated. During this phase, in situ data and laser ranging data will be collected from dedicated altimetric system verification sites to be used in the verification process. A verification workshop will be held five months after the beginning of the Verification Phase. The maximum duration of the verification phase is six months.

Science data products from the verification phase are reprocessed at the beginning of the observational phase using verified and calibrated algorithms.

The Observational Phase begins at the completion of the Verification Phase and continues to three years after launch. Instruments data are collected and processed systematically with a continuous instrument monitoring.

The Extended Observational Phase begins at the end of the observational phase and will continue at least two years until the end of mission five years after launch.

The sharing of operations between CNES and NASA/JPL depends on the phase of the mission. In term of operations, there are two major phases:

- the assessment phase, which begin after launch and continues until the in-flight acceptance of the satellite bus and the instruments.
- the routine phase, which covers the time after the launch + 2 months till the end of the mission life.

The sharing of operation functions between CNES and JPL is as follows:

- CNES JCCC will control the satellite from launch until the end of the assessment phase (2 months). JPL POCC will control the satellite after that time.
- CNES will perform satellite analysis, navigation, POD throughout the entire mission.
- CNES will generate and maintain the satellite database.
- CNES and JPL will provide science instrument analysis and command generation for their respective instruments throughout the mission.
- Only JPL will produce Operational Science Data Record.
- Both SSALTO and JSDS will produce altimetric science products in parallel.
- AVISO and PODAAC will provide the science products to their respective user communities.

The Jason operations documentation will be coordinated between CNES and JPL:

Operations Concepts document, Team Operations Plans, Team Interface Description, Sequence Element Dictionary, Training Plan, ...

Testing of the JGS will be performed in the following phases:

Unit Testing/Acceptance Testing: Each JGS element is responsible for testing at the element level. Each element is expected to develop a test plan and produce test reports. After the element development and internal integration is complete, each element will perform a set of acceptance tests to demonstrate that the element meets its requirements.

Technical Integration: During this phase, CNES and JPL will separately integrate their elements together into a system. This phase will concentrate on the capabilities needed for command and control of the Jason satellite platform.

Technical Qualification: During this phase, CNES and JPL elements will be integrated. The elements concerned with Payload command and control, science processing and sequence generation will be integrated into the JGS. The complete JGS will be tested end-to-end. The satellite to JGS interfaces will be tested using the actual satellite.

Operational Qualification: This phase will support the operations team training for all mission phases. Finally, a set of launch rehearsals will demonstrate that the JGS is ready to support the mission.

Status and Conclusion

The JGS detailed interfaces are defined between all the elements of the JGS. A Satellite Data Base (SDB) which describes all the telemetry and command structure is already existing. SDB will be filled up during next months and broadcasted to all JGS elements. Within the JGS, the NASA/JPL Earth terminals have been accepted in summer 1998, the CNES control center elements are being accepted (till February 1999), other parts (CNES Earth Terminal, JPL Control Center, CNES and JPL Mission Centers) should be completed by mid 1999.

The Validation and Qualification phases will take place during 1999.

The JGS qualification phases will be an important step in validating the JGS design and the concept for sharing operations between JPL and CNES. The close cooperation between CNES and JPL has been the major ingredient for getting an operational JGS with a high performances level.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 29-34

ISSN 0100-7386
Printed in Brazil

Design of China Space TT&C Network

Zhijian Yu
Baosheng Sun Zheng'an Zhai
Beijing Institute of Tracking and Telecommunications Technology (BITTT)
China Satellite Launch and Tracking Control General
P.O.Box 5131
Beijing 100094
P.R. China

Abstract

First of all, this paper summarizes the evolution of China space TT&C network in the past thirty years and experience accumulated. Then the design guidelines, network topology, composition, functions, major technical specifications and network operation management are also presented. Finally, issues relating to its future development are discussed and some ideas and considerations are given.

Introduction

With the development of China's space industry, the space TT&C network has gone through a process from small to big and from dedicated use to general application. A VHF/UHF satellite TT&C network and a C-band satellite TT&C network have been built for the TT&C of medium and low earth orbit satellites and geostationary satellites. The VHF/UHF TT&C network is near its end of lifetime. A new S-band space TT&C network consisting of mainly an S-band unified carrier system (USB) are under construction in recent years to meet the ever increasing demand of space TT&C requirements. It will provide support for most spacecraft of China as the new generation of space TT&C network. In the meantime, the new network is also capable of providing TT&C support for spacecraft of other countries and space organizations.

Tasks of the Space TT&C Network

The main tasks of China's space TT&C network are to replace step by step the VHF/UHF TT&C network and provide TT&C and communication services for spacecraft of China in medium and low orbit; to provide TT&C and communication services for China's S-band geostationary satellites; and participate in international inter-network connection and international cooperation and to provide TT&C services for other countries and space organizations over the world.

Design Principles of the Space TT&C Network

Design principles of the space TT&C network are as follows:

- Meet the TT&C requirements of common medium and low earth orbit spacecraft and geostationary satellites, including orbit determination accuracy, telemetry, telecommand and to provide a wide range of applicability in order to improve the utilization and economic efficiency of the whole TT&C network;
- The TT&C stations are based upon a microwave unified system with TT&C functions and their main technical specifications shall be state-of-the-art;
- Network architecture is adopted for data transmission and to provide flexibility for data exchange and convenience for users;
- Network management system shall be used for concentrated management of the TT&C stations and to improve the operational efficiency and automatic management level of the whole network;
- The network shall meet international standards to prepare for international cooperation.

Configuration and Distribution of the Space Network and System Design

China's S-band space TT&C network mainly consists of five USB stations, a network management center (NMC), a number of mission centers and a data exchange system consisting of mainly satellite communication means. The five USB stations are Kashi TT&C Station (KS), Weinan TT&C Station (WN), Qindao TT&C Station (QD), Xiamen TT&C Station (XM) and Nanning TT&C Station (NN). These stations can be divided into two classes. One class are Weinan TT&C Station, Qindao TT&C

Station and Xiamen TT&C Station which are used for TT&C of medium and low earth orbit spacecraft. The other class are Kashi and Nanning TT&C stations which can used for geostationary satellites as well as medium and low earth orbit spacecraft. The network management center is located at Xi'an Satellite Control Center (XSCC) and it is responsible for routine management of the whole TT&C network. Composition of the overall S-band TT&C network is illustrated in Fig. 1.

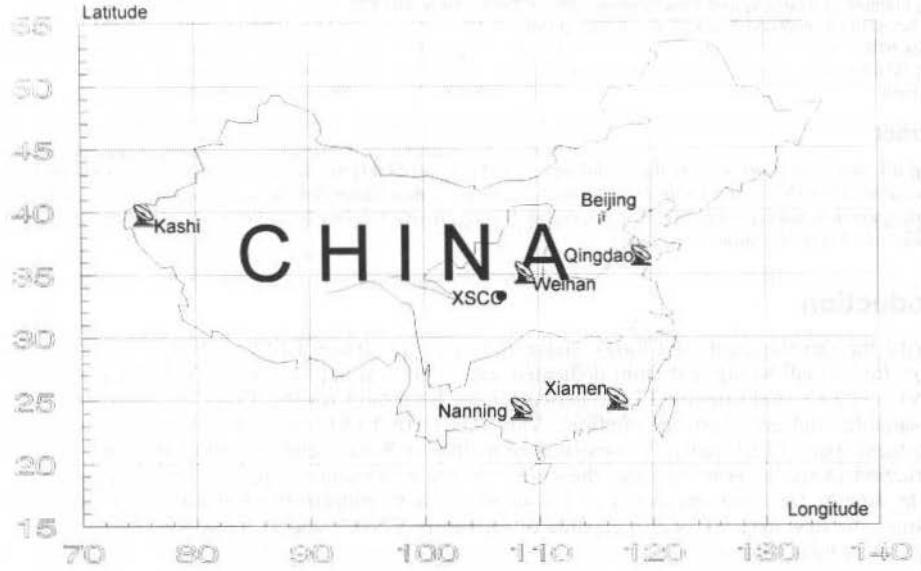


Fig. 1 Distribuition of China's S-band Space TT&C Network

USB stations in the network support TT&C and communication for medium and low earth orbit S-band spacecraft under the organization of the network management system. Distribution of the TT&C station within the network has considered the territory of China. Kashi TT&C Station, Qindao TT&C Station and Nanning TT&C Station are located in northwest, east and south China. Span of positions of the stations is optimized in latitude and longitude. This kind of station distribution is favorable in providing coverage for more passes of polar satellites and medium and low earth orbit spacecraft (3-5 passes everyday for sun-synchronous satellites). Xiamen TT&C Station and Weinan TT&C Station can provide redundancy for the above three stations and can also increase coverage . Kashi TT&C Station and Nanning TT&C Station can provide launch and early orbit (LEOP) services and TT&C support after positioning for S-band geostationary satellite under the organization of the Xi'an network management center. Nanning TT&C Station and Kashi TT&C Station have larger difference in longitude (around 32 degrees) and have long relayed coverage for geostationary satellites above the equator. They can provide services for S-band geostationary satellites within 10^0- 180^0E before and after positioning. The two stations can be redundancy for each other for satellites in the overlapped coverage of the them.

Function, Composition and Technical Specifications of the USB Stations

Main Functions of the USB Stations

The main functions of the USB stations of China include:

1) Performing TT&C missions using the unified carrier system, i.e.:

- Ranging measurement with a side tone system with the major tone of 100kHz;
-
- Dual direction Doppler range rate measurement;
- Angular measurement with dual channel monopulse system;
- Engineering telemetry reception and recording (data format compatible with ESA PCM telemetry standard);
- Transmission of command (data format compatible with ESA PCM command standard) and uplink data with two uplink subcarriers of the unified carrier system;

- Simultaneous reception of another S-band carrier with QPSK system reciever for high data reception;
- Concentrated monitoring of the whole USB with station monitoring and control console and report of the operation status of station equipment to the network management center under the remote control of the network management center.

Composition

As illustrated in Fig. 2, the USB stations consist of mainly tracking and angular measurement subsystem, auto-acquisition subsystem, transmit subsystem, receive subsystem, ranging subsystem, range rate subsystem, telemetry subsystem, command subsystem, high rate data transmission subsystem, uplink subsystem, recording subsystem, test and calibration subsystem, system monitoring and control console and DTE subsystem.

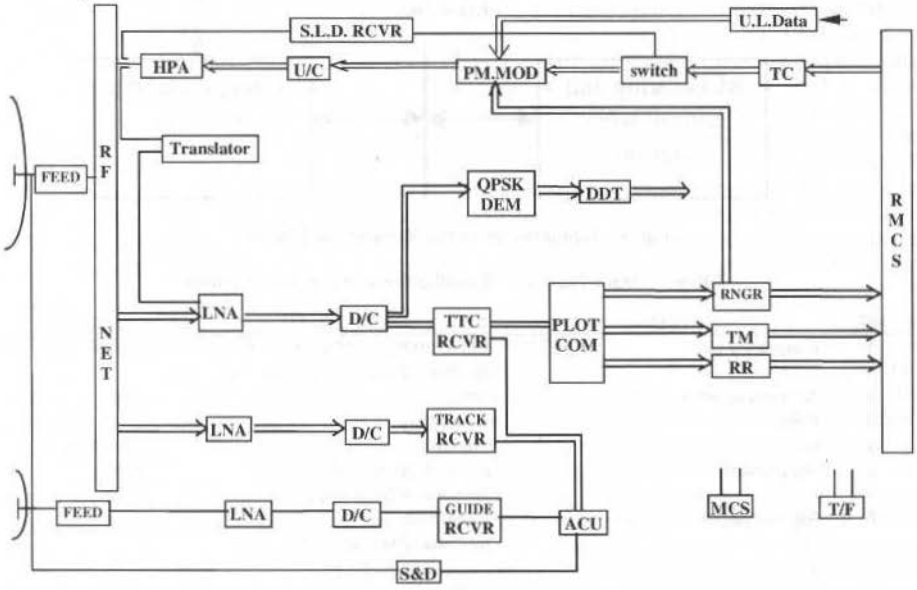


Fig. 2 USB Block Diagram

Main Technical Specifications

The stations in each class have the same specifications. The main technical specifications of the USB stations are given in Table 1.

Functions and Composition of the Network Management Center

Functions of the Network Management Center

The major functions of the network management center of the S-band space TT&C network include:

1) Reception of utilization requests from each mission center and work plan generation after combination and coordination, and concentrated planned management of each TT&C station within the network;
2) Remote configuration of equipment status of the USB stations in accordance with mission requirements. Configuration commands are sent to USB stations from NMC through data transmission system;
3) Concentrated monitoring of equipment status of each TT&C station in the network, alarm information and the configuration process of equipment status;
4) Concentrated control and status switching of the equipment of TT&C stations in the network.

Composition of the Network Management Center

As illustrated in Fig. 3, the network management center of China's S-band space TT&C network mainly consists of communication subsystem, front end processor, data processing server, monitoring and control work stations.

The communication system, mainly consists of satellite communication stations, terrestrial lines, timing, scheduling, telephone, telegram and telex, provide support for organization, command and data transmission and unified time reference for systems in the network management center.

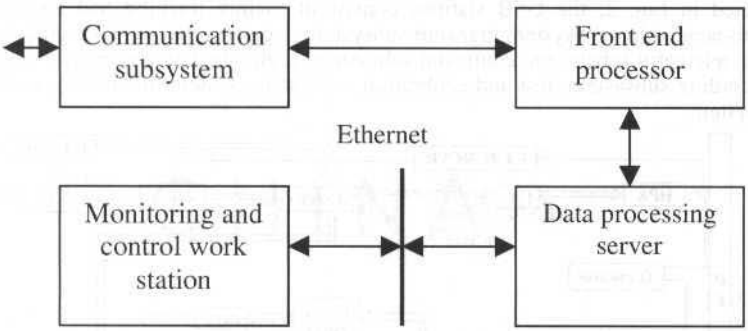


Fig. 3 Composition of the Management Center

Table 1 Main Technical Specifications of the USB Stations

NO.	ITEM	WN, QD, XM	NN, KS
1	Frequency band	Uplink: 2025 - 2120MHz Downlink: 2200 - 2300MHz	Same
2	Antenna aperture	10m	12m
3	EIRP	49 - 69dBW	51-71dBW
4	G/T	20.5dB/K	22.5dB/K
5	Polarization	Uplink: RCP or LCP Downlink: RCP and LCP	Same
6	RF modulation	Uplink: PM Downlink: PM (carrier 1) QPSK (carrier 2)	Same
7	Frequency accuracy	5×10^{-11}	Same
8	TC subcarrier frequency	2 - 100kHz	Same
9	TC subcarrier modulation	PSK, FSK	Same
10	TC subcarrier waveform	Sine	Same
11	TC data rate	100b/s - 2kb/s	100b/s - 8kb/s
12	Uplink data subcarrier Frequency	256kHz	Same
13	Uplink data subcarrier modulation	DPSK	Same
14	Uplink data subcarrier waveform	Sine	Same
15	Uplink data rate	32kb/s	Same
16	TM subcarrier frequency	2-512kHz	Same
17	TM subcarrier modulation	PSK, DPSK	Same
18	TM subcarrier waveform	Sine	Same
19	TM data rate	100b/s - 64kb/s	Same
20	Ranging tone waveform	Sine	Same
21	Major tone	100kHz	Same
22	Minor tone (kHz)	20, 16, (0.8, 0.16, 0.032, 0.008 on 16)	Same
23	Ranging error	15m	Same
24	Range rate error	5cm/s	Same
25	Angular measurement error	0.5mil	Same
26	QPSK downlink channel data rate	1Mb/s X 2	Same

The front end processor performs management and control, at link level and physical layer, the received and transmitted information; classification of all kinds of TT&C information received by the network management center and disseminate to mission control centers and USB stations according to network plan; send to the data processing server of the network management center information such as remote monitoring data of USB stations related to the network management center and utilization request of the TT&C networks of mission centers; receive information such as remote control commands for

USB from the data server of the network management center and echo of TT&C network utilization request, and send to USB station and mission centers.

The data processing server receives TT&C network utilization request from various mission centers, generates echo to TT&C network utilization request and TT&C network work plan, etc.

The monitoring and control work stations mainly provide display of operational status of USB equipment and send USB status configuration commands.

Operational Modes of the Space TT&C Network

Management Mode of the TT&C Network

Routine management and scheduling of China's S-band space TT&C network is performed by Xi'an Network Management Center. Spacecraft mission center send TT&C network utilization request to Xi'an Network Management Center in advance according to spacecraft TT&C plan. Xi'an Network Management Center performs concentrated coordination based on these utilization requests and the current status of TT&C network and forms TT&C network utilization echo and send to each user generating the request. The mission centers take the utilization request echo from the network management center as the basis for making work plans such as communication duration plan. Network management coordinates and generates TT&C equipment work plan and send the plan to each TT&C station. In the meantime, the network management center configures prior to missions the equipment of TT&C stations in accordance with the TT&C station configuration requirements of mission centers. The network management center combines the TT&C network utilization requirements of the mission centers and forms TT&C network scheduling plan for concentrated allocation and scheduling of TT&C network resources.

Monitoring and Control Modes of TT&C Networks

Space TT&C network monitoring and control can be divided into three levels, i.e., network management center, monitoring and control console of each USB system and subsystem front panel of each USB. Monitoring and control of USB is performed in a bottom to top order. Subsystems of USB collect, display the status of each subsystem and report to the monitoring and control console of USB system; the monitoring and control console of USB system collects and displays the status of each subsystem of its USB station and reports to the network management center; the network management center performs monitoring of the status of USB systems of the whole TT&C network. Items and contents for the monitoring at the three levels are not the same and decrease from subsystem to the network management center. The network management center cares most about the operation status at the system level and key subsystem level while each subsystem needs to know the detailed operation status of each plug-in element and components. During a mission, the mission center can also perform monitoring of equipment status of the stations used.

The network management center can perform configuration and control of USB in the network. Monitoring and control console of the USB systems can perform control of each USB. The monitoring and control front panel of each USB subsystem can perform control of each subsystem. The order of priority for USB status control is from top down: control front panel, system monitoring and control console, network management center; the order of utilization priority for USB configuration is: network management center, system monitoring and control console, front panel of each subsystem.

Transmission and Processing of TT&C Information

The processing of spacecraft TT&C information in China's space TT&C network is mainly performed by mission centers. USB stations send in real-time the metric information of R, $\dot{R}$, A, E and received telemetry information to mission centers. The mission centers perform processing and display of measurement information. Mission centers decide spacecraft status in accordance with the measurement information and make decisions for control of the spacecraft and send control commands to spacecraft through the information transmission system of USB stations. USB stations has a certain data processing capability of spacecraft telemetry and metric data and emergency transmission capability for selection based on different space missions.

Information Exchange of the Space TT&C Network

Information exchange of China's space TT&C network is mainly between TT&C stations and centers and between centers and centers. Main contents for the exchange include metric measurement, telemetry, commands, equipment status monitoring and control information, acquisiton information and network

work plan. Data transmission has two routes, one prime and one redundant and the data rate of both routes are 64kb/s. There are two lines of telephone and a dedicated circuit between the NMC and TT&C station for scheduling and technical coordination of the center on the stations. The main means of information exchange is satellite communications and the data transmission bit error rate is 1×10^{-6}.

Instrumentation information and data of the TT&C network is massive and very much varied while control information is required to be highly reliable. The main ways of data transmission are satellite communication . Frame relay protocols, which has large capacity, high bandwidth utilization efficiency with good real-time performance, is adopted for transmission of the bottom layer of TT&C network information based on the features of the information transmitted, the ways information is transmitted and current status of data transmission technology while TCP/IP protocols, which allow inter-connection between different computers and different networks, is adopted for higher layers (network layer and transmission layer). Connectionless (seamless) UDP protocols in TCP/IP protocols, which has broadcasting capability, high transmission efficiency and good real-time performance, is used to transmit downlink (TT&C station to the centers) information and those uplink (the centers to TT&C station) information which has lower reliability requirements. Connection-oriented reliable transmission protocols TCP, which has retransmission if error control function, is used to transmit uplink command information, etc., which has high reliability requirement. CCITT X.25 interface is also provided at the network management center for inter-connection with other TT&C networks, TT&C station and control centers.

Technical Features of the Space TT&C Network

China's space TT&C network has the following technical features:

1) High utilization efficiency.

TT&C network resources are managed by the network management center in request-echo-planning mode. Mission centers, the network management center and USB station are designed for multi-mission, they can be used for the TT&C of medium and low earth orbit spacecraft as well as geostationary satellites.

2) High level of automation and operational efficiency

USB station monitoring, control and equipment status configuration per mission in the network are all performed by the network management center.

3) High reliability and flexibility

Processing and generation of space TT&C information is usually done by mission centers while the TT&C stations also have data processing and emergency command transmission capability to a certain extent.

4) Extended capability

The TT&C stations have not only conventional TT&C functions, but also the capability to receive high data rate spacecraft data such as payload data.

5) Standardization

Design of the network meets relevant international standards and is compatible with the TT&C networks of other space organizations. This is very helpful for international network inter-connection and mutual support between China and other space institutions in the world.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 35

ISSN 0100-7386
Printed in Brazil

Constellations & Space Operations

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 36-43

ISSN 0100-7386
Printed in Brazil

Conceptual Leo Satellite Constellation Design

Shinichi Ishikawa
Ryutaro Suzuki
Tateo Goka
Yasuhiko Yasuda
Next-generation LEO System Research Center
National Space Development Agency
Waseda University
Telecommunications Advancement Organization of JAPAN
ishikawa@nels.tao.go.jp

Abstract

In designing a satellite constellation, many factors have to be taken into account including transmission delay, service coverage, minimum mobile terminal elevation angle and space radiation effect. In our project, LEO constellation with Inter-Satellite Links(ISLs) is assumed from the point that the low delay is one of the key factors of successful multimedia communications services. The LEO constellation parameters such as orbital altitude, orbit inclination, number of satellites, number of orbital planes are discussed in this paper. For example, the effect of Van Allen radiation belts and the South Atlantic Anomaly(SAA) is considered in choosing the orbital altitude and the orbit inclination. The constellation parameters are also investigated from the point of ISL configuration.
Keywords : *LEO, constellation, Van Allen, SAA, ISLs*

Background

In Japan , the Ministry of Posts and Tele-communications(MPT) conducted a feasibility study in 1996 on the Next-generation LEO System(NeLS) which would provide a global multimedia mobile satellite service by means of a group of LEO satellites. The target year for commercial implementation is 2010. In order to develop the feasibility study, NeLS Research Center was formed by the Telecommunications Advancement Organization of Japan (TAO) in cooperation with the telecommunications operators, manufactures, universities and governmental research organization in the end of 1997. The conceptual satellite constellation design and the development of key technologies needed to construct this Next-generation LEO system started from 1998. In this paper we deal with the conceptual satellite constellation design.

Constellation design process

A layout of the constellation design process is shown in Fig.1.

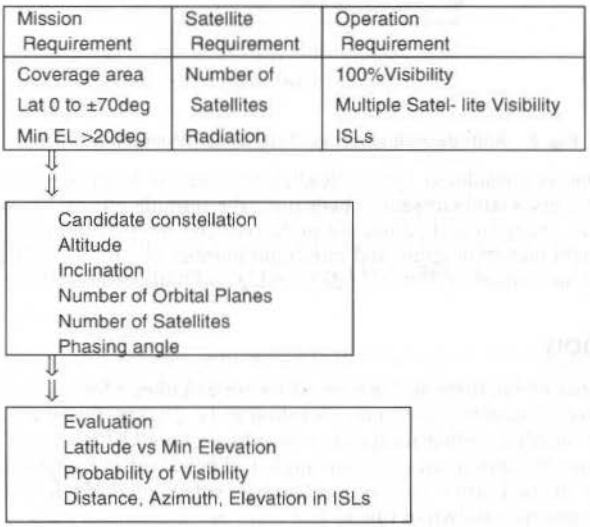


Fig.1 A layout of the constellation design process

At the first stage of the constellation design, the related mission objects and system description need to be clarified. From the information related to the mission (R. Suzuki et al.), we summarized the requirements in terms of a satellite constellation. When rearranging the requirements, it becomes apparent that certain requirements dominate the design process. Those requirements and the discussion to select constellation parameters are described in the following sections.

Radiation

The selection of orbital altitude and inclination as LEO is discussed here from the point of avoiding the space radiation effect.

The SAA is a portion of the Van Allen belts. The Van Allen belts are not a uniform distance from the surface of the Earth. SAA extend much closer to the surface. These belts extend from about 150km in altitude to about 930Km in altitude. This region is known as the SAA, in longitude from 50deg.west to 15deg.east and in latitude from 30deg.south to 50deg.south[2]. When the satellite passes through the SAA, Single-Event Upsets(SEU) occurs occasionally. SEU is an one of serious radiation problems as well as the total dose. To avoid the effects of SAA, the altitude were desired above 1000km.

The Van Allen radiation belts are two toroidal belts of charged particle radiation trapped by the magnetic field, centered on the equator. Such radiation gives two types of effects on spacecraft electronic systems. These are the degradation due to total accumulated dose and the malfunctions induced by SEU and Single-Event Latch-up(SEL). Both of them affect the fundamental survivability of spacecraft.

Since radiation levels depend on the orbital altitude and inclination, the selection of the orbital parameters is made from the point of minimizing these two types of radiation effects. The relation among the radiation level total accumulated dose using an aluminium shield of 100mil, altitude from 700km to 1,300km, and the inclination from 35deg to 75deg, is shown in the Fig.2.

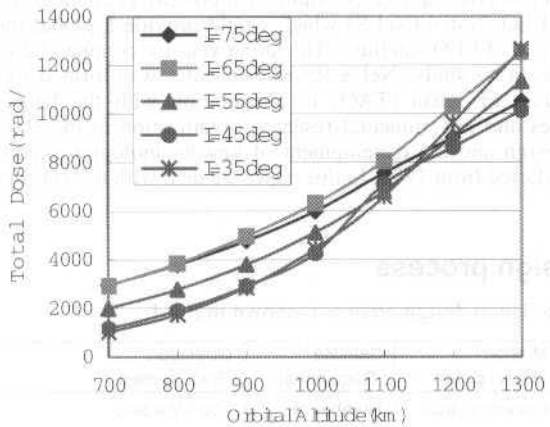


Fig. 2 Altitude/Inclination vs. Total dose (Aluminium 100 mil)

Our mission life time is considered 5years. Radiation hardness level for commercial semiconductor devices is considered 5years × 10^4 rad/year. Therefore , the altitude can be chosen under 1,200km and 75deg, 55deg, 45deg and 35deg in inclination are preferred. But higher altitude is desired because of the requirements for minimum elevation angle and minimum number of satellites. The orbit parameters with altitude of 1,200km and inclination of 75deg, 55deg, 45deg and 35deg were selected.

Minimum Elevation

The necessary number of satellites and that of orbits are calculated for a specified value of minimum elevation angle. Here, we assume the minimum elevation to be 20 deg. In the case of circular orbits, the service area of a single satellite, called footprint, is a spherical area of the earth's surface in which the satellite can be seen above the minimum elevation angle ε. The extent of footprint is determined by ε and orbital altitude h, radius of the Earth r(=6378km). The half-sided center angle ϕ of the footprint is given by equation (1). The geometry is shown in Fig.3

$$\phi = (\pi/2) - \varepsilon - \arcsin\{(r \cos \varepsilon) / (r + h)\} \quad (1)$$

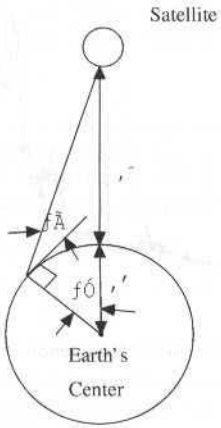


Fig. 3 Datellite coverage geometry

To cover the Earth's surface of the service area, it is inevitable that the footprints overlap. The largest possible effective footprint of a single satellite is equivalent to the largest hexagon inscribed into the footprint (M. Werner et al.). Fig.4 shows the hexagon inscribed into the footprint in the polar orbit.

Figure 4 shows that at least N orbits are necessary to cover the great circle along the equator. To cover the great circle along an orbit track S satellites per orbital plane are necessary.

$$N = 2\pi/3\phi \quad (2)$$

$$S = 2\pi/\phi\sqrt{3} \quad (3)$$

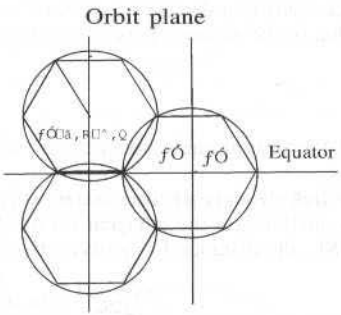


Fig. 4 Hexagons inscribed into the footprint

In the case that the minimum elevation angle ε=20deg, orbit altitude h=1,200km, and the orbit is polar, the value ϕ,N, and S are obtained using (1),(2) and (3) as follows.

$$\phi = 17.7deg,\ N = 7,\ S = 12$$

For the inclined orbit constellations, equation (3) does not give the optimal solution, because the concept of continuous coverage with hexagons along the orbit tracks is not applicable. But equation (3) can be used as a rough approximation.

Service Area coverage

To estimate the necessary number of satellites and orbits and to discuss the candidate constellations the service area coverage is one of the most important requirements. Figure 5 shows the world population

density by latitude (D. Diekelman). It can be deduced that area from –60deg to +70deg in latitude should be covered.

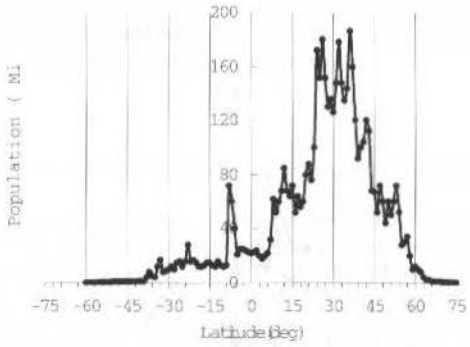


Fig. 5 World Population Density

Inter Satellite Links

The constellation parameters are investigated from the point of ISL configuration. The operation of both the intra-plane and inter-plane ISL are assumed. If a circular orbit is employed, the link distance and pointing angle between neighbor satellite are constant for the intra-plane ISL. As for the inter-plane ISL, the relation between a satellite and its neighbor on the adjacent plane is time-variant but simple because these satellites on different planes move synchronously. In this regard, the circular orbit or near orbit is preferable to the elliptical orbit.

The assumed ISL requirements are as follows.

- Intra-plane ISLs per satellite :2
- Inter-plane ISLs per satellite :2
- Link Distance : < 5,000km
- Angular coverage : - 90deg to + 90deg Azimuth
- 30deg Elevation

The requirement of link distance leads to a discussion of number of orbital planes and satellites. The distance D of the intra-plane ISL, that is, the distance between two adjacent satellites on the same plane is expressed by,

$$D = 2 (h + r) \sin (\theta / 2) \qquad (4)$$

where h is the orbital altitude, θ is the central angle between two adjacent satellites on the same plane, and r is the Earth's equatorial radius.

As for the inter-plane ISL, the link distance is time-variant. Fig.6 shows the parameters needed to describe the relation between two satellites on the adjacent orbital planes. As an approximation of the maximum distance of inter-plane ISL, the distance D' between two ascending nodes of adjacent orbital planes. That is,

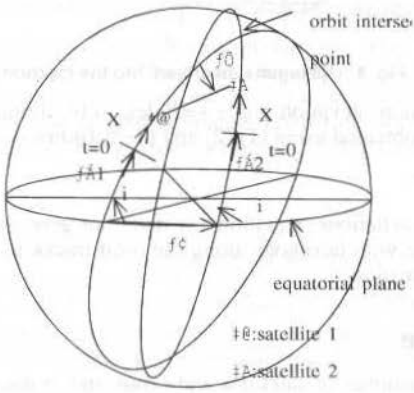


Fig. 6 Geometry of two satellitew

elevation requirement of 20deg at latitude from ±45 to ±70deg , from –60 to –70deg, and from –15 to 15deg, respectively

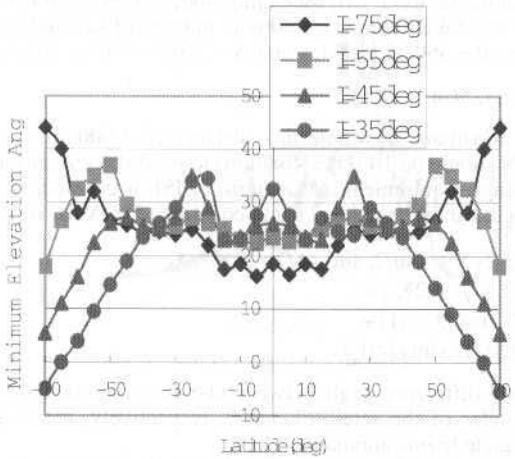


Fig. 8 Minimum Elevation Angle vs. Latitude

Fig.9 shows probability of visibility(POV) vs. minimum elevation over the region from –70 to 70 deg in latitude.

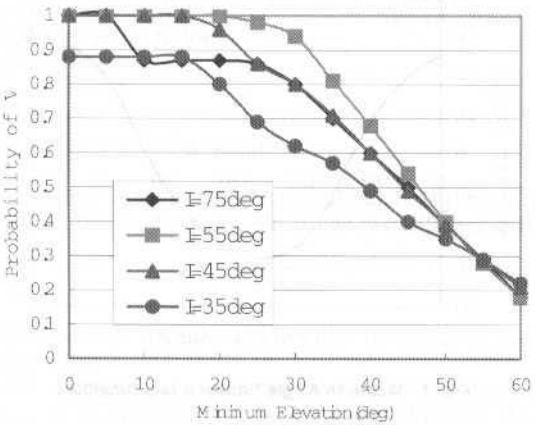


Fig. 9 Probability of Visibility vs. Minimum Elevation

Table.1 is obtained from the same simulation. These figure and table show that the constellation of I = 55deg gives better POV than that of I = 35deg, I =45deg or I = 75deg. Then, we selected I = 55deg.

Table 1 Comparison of minimum elevation

	100% POV	95% POV
I = 75deg	16deg	22deg
I = 55deg	19deg	28deg
I = 45deg	3deg	7deg
I= 35deg	none	none

For the constellation of I = 55deg, further evaluation is done with regard to the ISL configuration and multiple satellite visibility.

Fig.10 and Fig.11 show the pointing angle variation in elevation and link distance variation, respectively, for inter-plane ISL in the constellation of I = 55deg.

A multiple visibility is a significant property to achive a given quality communication service in case of a satellite failure. Table.2 shows the comparison of minimum elevation of single, two, and three satellites simultaneously.

$$D' = 2 (h + r) \sin (\Delta/2) \qquad (4)'$$

where Δ is the difference angle between two ascending nodes of adjacent orbital planes. If, as discussed in the previous section, the orbital altitude h = 1,200km, number of satellites per orbital plane S = 12 and number of orbits N = 7, then $\theta = 360/12 = 30$deg and $\Delta = 360/7 = 51$deg. Therefore, from (4) and (4)'

$$D = 3922km \text{ and } D' = 6525km$$

Since D' exceeds the maximum allowable link distance of 5,000km, N = 7 orbits are not enough. From (4)', it is found that N should be 10 (D' = 4683km) to meet the link distance requirement.

Next, the angular coverage requirement(in Azimuth) of ISL is considered. The azimuth angle ψ from satellite1 to satellite2 in Fig.6 can be calculated using equation (5) (A. H. Ballard).

$$\begin{aligned}\psi = &[\sin i \sin(\Delta/2) \cos(\gamma 2 + X) - \sin 2i \sin^2(\Delta/2)] \\ &/ [\sin^2 i \sin^2(\Delta/2) \sin(\gamma 1 + \gamma 2 + 2X) + \\ &2\cos i \cos(\Delta/2) \sin(\Delta/2) \cos(\gamma 2 - \gamma 1) + \\ &\{\cos^2(\Delta/2) + \cos^2 i \sin^2(\Delta/2)\} \sin(\gamma 2 - \gamma 1)] \end{aligned} \qquad (5)$$

Here, i is inclination, Δ is difference angle between two ascending nodes of adjacent orbital planes, $\gamma 1$ and $\gamma 2$ are initial phase angles of the satellite1 and 2, respectively, measured from the point of right ascension, and X is phase angle (mean anomaly).

Fig.7 shows the azimuth angle between two satellites which co-rotates each other in the case of I=55deg. The variations of the azimuth angle are from –55deg to 55deg which is within the requirement stated above. In this case, we assume that the phasing angle = 3deg, N= 10 orbits, S = 12 satellites per orbit.

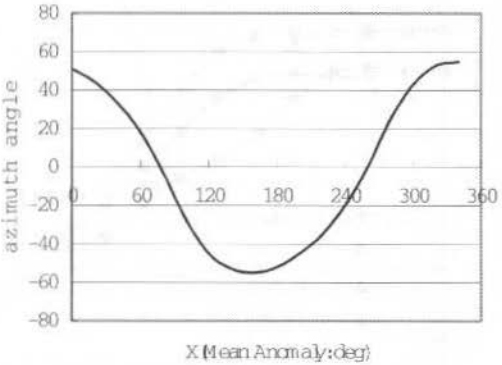


Fig. 7 Azimuth Angle Between two Satellites

Candidate Constellations

The candidate constellations obtained from the above discussions are as follows.

- Number of orbital planes : 10
- Number of satellites per a orbital plane : 12
- Orbital altitude : 1,200km
- Eccentricity : 0
- Inclination : 75deg, 55deg, 45deg, 35deg
- Difference angle of ascending node between adjacent orbital planes : 36deg
- Phase angle : 3deg

Evaluation

The computer simulations using STK and Satlab were performed to check whether the candidate constellations meet the above requirements. Fig.8 shows the minimum elevation angle vs. latitude. The figure shows clearly that the constellations of I=35deg, 45deg and 75deg do not meet to the minimum

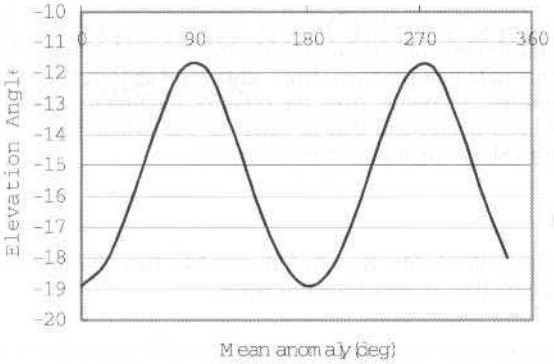


Fig. 10 Elevation Angle Between two Satellites

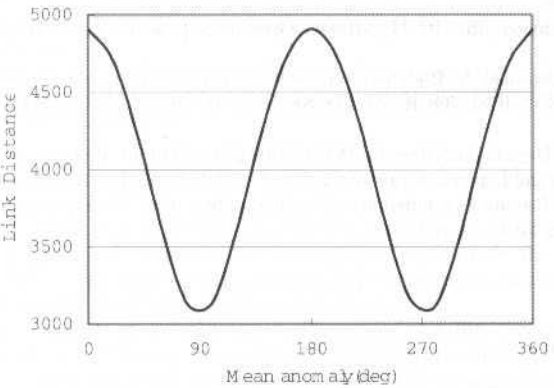


Fig. 11 Link Distance Between two Satellites

Table 2 Comparison of minimum elevation

	100% POV	95% POV
1 satellite	19deg	28deg
2 satellites	15deg	22deg
3 satellites	8deg	13deg

Fig. 12 shows the ground track of the following constellation

- Altitude:1200km
- Inclination:55deg
- Eccentlicity:0
- Number of orbital plane:10
- Number of satellite per orbital plane:12
- Phase angle:3deg

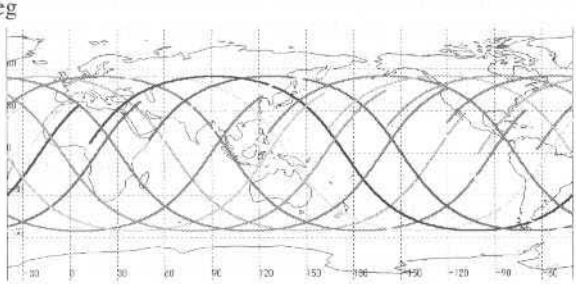


Fig. 12 Ground track of the selected constellation

Conclusion

Several factors were taken into consideration in this paper for the constellation design of the NeLS. The first factor is the space radiation problem, from which the appropriate range of orbital altitude and orbit inclination was identified. Secondly, from the point of minimum mobile terminal elevation angle, the necessary number of orbit planes and that of satellites per orbital plane were discussed. Then, the constellation suitable to the ISL configuration was examined.

Acknowledgment

The authors would like to thank Mr. Keiichi Sakurai, Mr. Iwao Nishiyama, Mr. Hirotaro Ooba for their much assistance and criticisms in preparing this paper.

References

R. Suzuki, K., Sakurai, S., Ishikawa. and Y., Yasuda. "A Study of Global Multimedia Mobile Satellite Communication System", IAF-98-M.3.06.

Attitude and Pointing Handbook, JSC-10511, Mission Operations Directorate, Revision B, JSC, NASA, February 1988.

M. Werner, A, Jahn, E , Lutz and A. Böttcher, Fer. 1995, "Analysis of System Parameters for LEO/ICO-Satellite communication Networks", IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, VOL.13, NO. 2,February 1995.

D. Diekelman. Nov. 1997, "Design Guidelines for POST-2000 Constellations, Proceeding at an international workshop entitled Mission Design and Implementation of Satellite Constellation", Toulouse, France.

A. H., Ballard, Sep. 1980, "Rosette Constellations of Earth Satellite, IEEE Transactions on Aerospace and Electronic Systems". Vol. AES-16, No 5.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 44-48

ISSN 0100-7386
Printed in Brazil

Operational Concepts for Orbit Control of Leo Satellite Constellations

Nathalie Dubernet
Jean Dulac
CNES, Centre Spatial de Toulouse
18, avenue Edouard Belin
31401 Toulouse Cedex 4, France
nathalie.dubernet @cnes.fr
jean.dulac @cnes.fr

Abstract

The increasing number of satellite constellation projects requires a new approach for design, development and operation to optimise facilities and reduce costs. The new problems posed by controlling satellite constellations are related to group launches during the deployment phase and to the large number of satellites to be controlled during their operational phase. From an operational point of view, the CNES studied new operation concepts, based on some hypotheses on the design of the satellites (autonomy, homogeneity) and the significant number of repetitive tracking and control tasks that this will involve.

By making some hypotheses on the satellite design, this paper aims at explaining that it is possible to imagine operation concepts both for deployment phase and on station operations leading to acceptable costs.

***keywords**: Constellations, operations, satellites.*

Introduction

For several years CNES has been involved in satellite constellation studies, both theoretical, through several R&T studies, and practical, in particular through participation in the STARSYS and then the SKYBRIDGE projects. This paper presents our preliminary conclusions on operation concepts for orbit control of satellite constellations dedicated to telecommunications.

Telecommunication systems are now being built around satellite constellations: IRIDIUM (66 operational satellites) has been completely deployed, the first satellites of the GLOBALSTAR system (48 operational satellites) have been launched, while the SKYBRIDGE system (80 operational satellites), and the TELEDESIC system (288 operational satellites) have been announced for around 2001/2003. We have deliberately restricted our study to constellations having common characteristics, namely:

- a telecommunication mission,
- small satellites with a mass of between 400 and 1200 kg,
- circular 'low' orbits of between 800 and 1500 km in altitude.

The extension of the conclusions of our study to other types of constellations (micro-satellites, orbits with an altitude of more than 10,000 km), should be carefully analysed.

The new problems posed by controlling satellite constellations are related to multiple launches during the deployment phase and to the large number of satellites to be controlled during their operational phase:

- for obvious economic reasons the deployment phase should be done as quickly as possible. This requires multiple launches (of up to 10 satellites simultaneously for example), at a high rate, which in turn, raises problems concerning the first acquisition and critical manoeuvres for several satellites at the same time.
- so that operating costs remain within 'reasonable bounds', the concepts implemented for simultaneous control of a family of 3 to 4 satellites, which is currently the case for CNES, cannot easily be extrapolated to cover control of constellation of more than 20 satellites.

The concepts described below are based on some hypotheses on the satellite design. These hypotheses are described in the first section. We then describe the operational concepts during the operational implementation phase (station keeping), followed by complements for the deployment phase.

Satellite design hypotheses

The LEO (Low Earth Orbit) satellites controlled by CNES require relatively significant human resources and other facilities:

- The lack of onboard autonomy requires several onboard/ground contacts for control manoeuvres and localisation measurements.

- The complexity of onboard operating modes requires tricky operational procedures. For instance, the return from safehold mode may require up to one week of critical operations.

It is obvious that the operating costs needed to control a constellation with the same constraints for each satellite of the constellation would be extremely high.

The hypotheses described below should make it possible to reduce ground operations and therefore the operating costs. They are mainly based on greater autonomy of the satellites with respect to the ground segment, and on simplifying operations by means of more efficient distribution of control functions between the onboard and ground segments.

Satellite autonomy

Specific case of first acquisition

In order to reduce deployment time and reduce launch costs, satellites are launched in clusters.

The first acquisition stage (first contact with the satellite after separation from the launcher) is particularly critical mainly because of possible launcher deviation, the risk of anomalies due to the launch and emergency initialising operations to be performed on the satellite.

Launching satellite clusters increases the criticality of this stage due to the number of operations to be performed almost simultaneously.

Following separation from the launcher, onboard/ground contacts have to be made with each satellite to ensure that it is operating correctly. In the case of a cluster launch this operation may take some time due to orbit uncertainty following separation and the number of satellites to be dealt with.

Therefore the satellites should be autonomous for the initialisation phases (pointing, deployment of solar arrays). Following separation, each satellite should autonomously switch to a safe mode enabling it to wait for the first onboard/ground contact. From between 12 to 24 hours after the separation can be necessary to achieve this stage.

For all stages

As already said, currently controlled LEO satellites require a significant number of onboard/ground contacts for localisation purposes and constraints related to satellite autonomy (excluding mission constraints).

The number of daily onboard/ground contacts required must be reduced in order to optimise the ground facilities used for controlling a constellation. This may be achieved:

- by using onboard facilities for localisation measurements. The two means available are the GPS (Global Positioning System), a constellation of 24 satellites in MEO orbit (20200 km in altitude, 55 ° inclination) transmitting signals in L-band and DORIS (Doppler Orbitography and Radio-positioning Integrated by Satellite) , a network of 53 beacons distributed over the Earth's surface and transmitting in 2 frequencies (400 MHz, 2GHz).
- by reducing the number of onboard/ground contacts dedicated to satellite control. The ESA standard is one of the means used to achieve this goal. It enables, in particular, an appropriate choice of telemetry parameters to be sent according to the mission stage. This makes it possible to limit the number of onboard/ground contacts by sending only that information which is relevant for control purposes.
- by possibly using inter-satellite links such as those used to control satellites in the IRIDIUM constellation.

Simplifying operations

The complex operating modes used for current satellites affect operations in two main ways:

- they require implementation of complex operational procedures
- they entail a risk of operational error.

The move to a constellation with this kind of satellite would be even more risky in terms of operations.

Satellites making up a constellation should therefore be simpler to operate thanks to simplified monitoring and command concepts. This may be achieved by :

- generalising the use of macro-commands.
- simplifying, for instance, the operations necessary for leaving the safehold mode.
- having identical operational procedures (or very similar ones) for all of the satellites, assuming the homogeneity of the satellites included in the constellation.
- defining, *a priori*, a strategy for abandoning faulty satellites to avoid prolonging unnecessarily the life of a given satellite (which is very costly in terms of human resources).

- limiting the impact of temporary loss of one of the satellites in the constellation on the other satellites, by setting up a robust constellation.

The main satellite concepts to be noted are thus: more satellite autonomy and simpler operating modes. The following sections will deal with the principal ground-based operational concepts to be implemented using these satellite concepts.

Operation concepts for the orbital operational stage (station-keeping)

The concepts described here are based on CNES experience with orbit control of LEO satellites.

Using the satellite design concepts described above, it is possible to consider ground operation concepts suitable for constellations. This section summarises recent CNES study in this field.

The concepts are mainly based on:

- automation of operations for control of healthy satellites so that human resources may be reserved for satellites in difficulty.
- comparing satellites in order to anticipate anomalies.

Reinforced automation for routine satellite control

The following routine tasks may be automated: daily monitoring/ commands, payload uploading, orbit monitoring and manoeuvre computation, production of reports and logs.

After a qualification period and if no particular problems occur, no human intervention should be necessary. In routine mode, following processing of the acquired telemetry during a pass, the control computer system should generate a brief status report for each satellite (good or bad) and may, in the event of an anomaly arising, automatically alert the personnel on duty.

Under these conditions, no operator should be necessary for controlling the platform. However if the mission availability requirements are very demanding (which is often the case for telecommunication mission) an operator may be required at all times.

This automation of routine tasks should make it possible to greatly reduce the operational workload for healthy satellites and to focus all effort on satellites which require specific attention (satellites in a critical stage).

Managing of faulty satellites

Special efforts in terms of facilities and human resources are needed to manage a faulty satellite. This is done by one or several engineers using a work station providing them with real time or stored telemetry.

For controlling satellites in a constellation, in order to obtain just the right number of facilities and human resources, limits should be set for managing faulty satellites. This may be done by:

- limiting the number of faulty satellites which can be simultaneously handled,
- defining, *a priori*, the criteria for stopping investigation of an anomaly and abandoning a satellite in difficulty.

Comparing satellites

Exploitation of a large number of practically identical satellites means that it is possible to compare the behaviour of various satellites in order to detect satellites which are acting differently and thus anticipate possible anomalies.

Operational concepts for deployment

The operational concepts referred to in this section concern only deployment operations.

The main constraint related to deployment operations is the overall time needed to deploy the entire constellation. This implies very closely staggered multiple launching. The workload for these operations depends on the deployment plan, with significant peaks during launch phases. For better distribution of the workload, the deployment strategy should take operational constraints into account to the extent that this is possible.

In particular, critical operations such as manoeuvres and in orbit tests should be planned to avoid overlapping with launches (which are critical for first acquisition).

The deployment of the constellation, cluster by cluster, one at a time, is broken down into three main phases:

- first acquisition of the satellites of the cluster after launch,

- orbit positioning operations (manoeuvres, drifts, configuring operations), which begin after the first acquisition and finish when the satellite has been placed on its operational orbit.
- In orbit test (IOT) operations.

First acquisition

This is a very critical phase because:

- it is the first contact with the satellite after its separation of the launcher,
- the final launcher deviation is not predictable.

The purpose of this phase is to make rapid contact (within a few hours) with all of the satellites in the cluster in order to check on board initialisation results and acquire the after launch orbit.

Given the uncertainty of the satellites orbit (possible launcher dispersions), the first stage involves acquiring a limited number of carefully chosen satellites (for instance from different dispensers), to determine the actual orbits achieved by the launcher.

Once this information has been obtained, the positions of the other satellites are deduced, thus facilitating their first acquisition.

This first acquisition strategy requires sufficient onboard autonomy (see satellite concepts).

Orbit positioning

There are two types of operation for orbit positioning :

- satellite operations which are called critical, and which require real time operations by satellite experts (manoeuvres, monitoring of specific satellites)
- operations which are considered to be non-critical and which do not require any specific operational constraints (routine monitoring).

Likewise, for station-keeping, the main thing is to allocate facilities (equipment and human resources) according to the criticality of the operations to be performed.

So-called non-critical phases are handled by automating operations so that human efforts can be focussed on satellites undergoing critical phases.

For non-critical phases, the satellite control concepts are explained in section 3 for the routine mode (automation of operations).

Critical operations require specific attention with due regard for safety.

Each satellite is controlled by a specially assigned operational team. This team monitors the satellite until it has been placed on its operational orbit. It consists of four to six people (satellite and flight dynamics engineers). The same team might be responsible for a group of satellites.

The total number of specially assigned teams depends on the maximum number of satellites to be managed simultaneously in critical phases. Consequently, in order to avoid a surfeit of operational teams, the number of simultaneous critical operations should be limited.

In Orbit Test operations

These operations require several onboard/ground contacts and the participation of satellite experts. They are thus very demanding in terms of operations. To reduce these operations and take advantage of the similarity between the satellites, test operations could be organised in the following way:

- in-depth testing and measuring of the first satellites
- more limited testing and measuring of the following satellites.

Conclusion

The aim of this paper was to describe a few concepts to achieve 'reasonable' operating costs for control of constellations of more than 20 satellites.

Our first conclusion is that this objective can only be achieved if the satellites in the constellation are sufficiently autonomous, in order to reduce the workload for ground operations and make them reliable. This involves three fundamental points:

- for localisation purposes, the satellites should be autonomous (using GPS or DORIS systems), in order to reduce the number of ground stations and the corresponding workload,
- all after launch initialisation stages should be as automated as possible to avoid unnecessary duplication of facilities and human resources needed for first acquisition of the satellites cluster,
- well-designed use of the ESA standard should make it possible to optimise satellite/ground contacts and hence reduce the operational workload.

There are two important points with respect to the constellation:

- the satellites in the constellation should be as homogeneous as possible so that generic procedures can be widely used,
- the constellation should be 'robust'. By this we mean that temporary or complete loss of a satellite in the constellation should not require complex and hence costly (in terms of operations) reconfiguring in order to maintain a mission.

Given these hypotheses, the following approach should be used to limit operating costs:

- a distinction should be made between 'healthy' satellites (those with a 'routine' status) and 'faulty' satellites (those in a critical stage),
- the processing of 'routine' status satellites should be automated as much as possible.
- the number of satellites in a critical stage simultaneously (as they are naturally the most demanding in terms of facilities and human resources) should be limited.

To satisfy the latter condition:

- during final orbit acquisition, critical operations on any one group of satellites should be limited (to 3 or 4 for instance), and this should be one of the specifications for mission analysis.
- only a limited number of faulty satellites should be handled at any one time. This implies that criteria for abandoning faulty satellites should be defined.

If the hypotheses on the satellites design are valid, then the concepts we have just formulated should make it possible to achieve the goal of relatively similar operating costs for a constellation of more than 20 satellites as for current costs for a 4 to 5 satellites family such as the ones developed in last 15 years.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 49-56

ISSN 0100-7386
Printed in Brazil

Space System Toolbox for Satellite Constellation Design and Control

Veniamin V. Malyshev
Vladimir V. Bobronnikov
Mikhail N. Krasilshikov
Olga P. Nesterenko
Alexander V. Fedorov
Moscow Aviation Institute (MAI)
Volokolamskoye shosse 4, Moscow Russia
malyshev@glas.apc.org

Abstract

For the last several years the Satellite Tracking and Control Center of INPE (Brazilian Institute for Space Research) and the System Analysis and Control department of MAI have performed joint studies on constellation design and operation. The software package ***Space System Toolbox*** *is one of the results of this work. The package consists of 11application tools, global data base and operating shell. There are two groups of tools in the toolbox: tools for constellation R&D (4 tools), and tools for real time operations with existing constellations and ground facilities (7 tools). Major application area of the tools are LEO satellite constellations, especially the Earth observation satellite constellations. A general overview of the Space system Toolbox and descriptions of particular tools are presented in this report.*

Keywords: *Constellation, Design, Simulation, Navigation, Control, Mission Planning, Performance Analysis, Software Complex.*

Introduction

Problems of optimal design and effective exploitation of space systems using low and medium altitude satellite constellations for communication, navigation and the Earth observation purposes are of major interest in space science and technology.

To solve those problems successfully, it is necessary to develop corresponding means, methodology, techniques, algorithms and software complexes. The last one have to be able to store necessary input data and to solve particular application problems at the phases of R&D, deployment, control and replacement of a constellation, operational scheduling of on-board payloads and on-ground facilities for navigation and interaction with satellite payloads purposes.

The Satellite Tracking and Control Center of INPE and the System Analysis and Control department of MAI have performed joint work in this direction during last several years. One of the results of this work was the creation of the software complex *Space System Toolbox,* which can be utilized at different phases of a constellation development and utilization for various satellite systems but, first of all, for the Earth observation satellite systems.

Description of the complex and capabilities is given in this report.

Space System Toolbox Description

Objectives

Software tools developed for R&D and real time operations of the Earth observation satellite system are integrated in the *Space System Toolbox.* The toolbox consists of a multi-purpose database and 11 applications. The *Constellation Optimization* tool allows to optimize a constellation configuration. The *Mission a Priory Performance Analysis* tool is designed for decision making at the R&D, deployment, maintenance and replacement phases. The *Constellation Deployment* tool optimizes deployment strategy of a constellation. The *Constellation Dynamics Simulator* predicts orbits of a constellation with preset accuracy. The *Tracking Stations Scheduler* program generates operation timeline for ground tracking stations network. The *Multi Satellite Orbit Determination* program evaluates statistics of relative positions of satellites in a constellation. The *Constellation Control tool* computes constellation station keeping strategy. The *Orbit Control Scheduler* is designed to plan a sequence of orbit corrections to keep a satellite slot in a constellation. The *Operational Mission Planning* tool is used to form optimal timelines for on-board observation instruments and ground imagery data acquiring stations. The *Operational Mission Performance Analysis* tool computes a set of mission performance indexes for

specified period of a constellation operation. The *Flight Operations Plan Compiler* tool creates the flight operation plan pages.

Architecture

The Space System Toolbox integrates the Operating Shell, the Database and application tools (see Fig. 1).

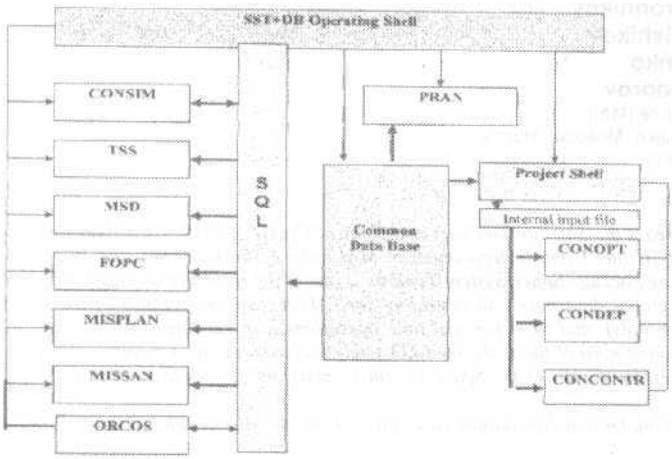


Fig. 1 General diagram of the Satellite System Toolbox

All the tools share common input data, and any application can be launched from one place. Local integration also is provided. Particular tools have local shells which create internal format of input requests to global database.

The *Operating Shell* is the core of the system. The shell has built-in database functions and applications loader. Database functions implement user-friendly intelligent interface. Applications loader runs any application registered in the shell. Any application validates input at the start, blocking processing of invalid input.

The Mission a-Priori Performance Analysis Tool

An objective of the Mission Performance a-Priori Analysis, or **PRAN** tool is evaluation of a mission performance for observation satellite system at the R&D and the long term (a-priori) mission planning phases. This information can be used to make decisions on the observation constellation design and to form an observation program.

The tool solves the following problems of a constellation efficiency analysis:

- analysis of known constellations to select an optimal one;
- constellation effectiveness analysis to derive constellation deployment program;
- long term analysis and performances prediction considering constellation distortions;
- effectiveness analysis for control of a constellation at maintenance and replenishment phases.

The PRAN tool provides analysis in respect to an *observation project* consisting of a set of *application tasks.* Each task is characterized by the following specification: ground region to observe; demanded ground spatial resolution; revisit time; band of spectrum; weight, or priority factor of the task.

Output data include parameters for decision making, including:

- *mission performance indexes:* common efficiency indexes of the mission, particular task performance indexes, statistics of a revisit time (main performance index of the system);
- mean number of observation tasks solved; and probabilities of all tasks solved; are the *common performance indexes;*
- *particular task performance indexes* include probability of the task fulfillment, probability of each requirement to the space imagey is satisfied.

Observation *revisit time statistics* are lower, upper and mean values, probability distribution function. The revisit time statistics can be estimated with respect to a particular region; an area observed with a

specified revisit time; regions to be observed in the particular band of the spectrum; regions to be observed with the specified ground resolution.

The **PRAN** tool includes three principal utilities: data retrieval, executive, and output display. The data retrieval utility provides access to the global database with the constellation identifier as input. The request is transformed into internal data files that are the input files to the executive utility. The executive utility generates internal output files that can be visualized by the output display utility.

The user is allowed to select a constellation to analyze (from the global database); enter new orbit parameters, observation instrument parameters, observation task specifications; edit all the groups of input data; browse results in tabulated form; browse results in graphical form.

The Constellation Optimization Tool

The Constellation Optimization, or CONOPT tool is designed to support the Earth observation constellation R&D, maintenance and replacement with the system performance evaluation for strategic decision making. The tool is also the kernel of constellation control and deployment routines.

A constellation is optimized for a given set of application tasks (see description of the PRAN tool) and available observation instruments. Optimization procedure consists of evaluation of ascending nodes of circular orbits and locations of satellites in orbits providing maximum of the chosen optimality criterion.

The following initial data and constraints are used for constellation optimization:

- *constellation geometry* (number of satellites, number of orbits, orbit altitude, inclination);
- *satellite payload* (view angle, spectral band, resolution);
- *observation project,* described by the set of specifications, similar to the given in the PRAN tool description.

Probability that requirements of the observation project are satisfied is considered as the optimality criterion. Optimization algorithm provides choosing the Walker's type constellation. Analytical technique for evaluation of revisit time statistics for specified ground regions are used.

Variants of constellation geometry are generated for a given set of initial data and the optimality criterion is computed for each variant. Optimal solution is selected over the variants, providing the best criterion value.

The CONOPT kernel is the procedure, which evaluates the criterion for an arbitrary investigated constellation for observation project at input. This permits to use that kernel for other applications. The CONDEP tool calls the CONOPT to return optimal parameters of a partially deployed constellation. The CONCONTR tool uses CONOPT to figure out the constellation correction moments, considering violation of revisit time requirements.

Two options of input data can be used for the CONOPT tool run: local file and global database. The first input option allows using the tool as a standalone application to examine different constellations at R&D phase. A design solution then can be sent to the global database. The second option allows for decision making on the constellations performed by the Mission Control Center as a part of its operational activity.

The Constellation Deployment Tool

The Constellation Deployment, or **CONDEP** program optimizes long term deployment strategy of the Earth observation constellation. Optimization consists of choosing a satellite launching timeline considering the constellation performance time history as well as capabilities of the launch complex. This program also can be used to define an appropriate launcher for constellation deployment. Two types of deployment strategy can be analyzed and optimized using the tool. The first one assumes that satellite are inserted into nominal slots, or *fixed deployment strategy.* The second one generates an intermediate constellation geometry which is optimal for satellites already in orbit and due for launch (so called *flexible deployment strategy,* assuming possibility of repositioning of the constellation satellites after each new satellite insertion). The class of Walker's constellations with uniformly distributed satellites is considered.

Computations are conducted with the following input data:

- *constellation geometry* (orbit altitude, inclination, number of orbits, number of satellites in orbits);
- *satellite and observation instrument performances* (propellant for orbital maneuvers, instrument view angle, spectral band, resolution);
- *observation project,* with characteristics given in the PRAN tool description;

- *launch complex characteristics* (time interval between launches, number of satellites per single launch).

Optimal deployment strategy is figured out from maximization of a constellation optimality criterion with respect to controllable parameters at a given deployment phase. Probability of the event, that requirements of the observation project are satisfied, is considered as the optimality criterion. Optimization algorithm is based on step-by-step solution of partial problems using exhaustive search as well as direct search procedures. No constellation distortion due to external factors is considered. Flexible deployment procedure uses Hohmann's transfer between circular orbits to evaluate characteristic velocity for repositioning.

The program loads constellation parameters and observation project attached to the constellation. User can choose the deployment strategy type and local options. Output of the program is stored in the report file. It contains the timeline of launches, the constellation geometry and performance time history, the propellant load over all constellation deployment period.

The Constellation Dynamics Simulator Tool

The Constellation Dynamics Simulator, or CONSIM program propagates orbits of a constellation for a specified period with the preset accuracy, considering the Earth's gravity field, atmosphere drag, Sun and Moon gravity and solar radiation pressure. An orbit is propagated by the Dorman-Prince numerical integration method. The results are approximated by Chebyshev's polynomials of the specified order.

CONSIM input consist of constellation configuration, propagation interval, disturbing factors to consider, parameters of mathematical models of disturbing factors, allowable error of numerical integration, Chebyshev's expansion order and subintervals for approximations.

Optionally, it is possible to select external disturbances to consider impact of:

- the Earth's gravitation field model (up to the 50th order);
- the Sun and the Moon gravitation (according to IERS);
- atmospheric drag (according to Standard Upper Atmosphere Model),
- solar radiation pressure considering penumbra.

CONSIM consists of the Service Database Module (SDM), the Executive Module and the Orbit Viewer. The SDM retrieves constellation geometry and satellite characteristics from the Global Database upon the request for the constellation to propagate. Then the user creates a Propagation Query. To track the systems already in service, the *Operate Database* is created and managed by the SDM. The *Operate Database* contains the following information: space system common name, slot code of the space system; orbit code; slot status; date and time instants of the propagation beginning and end; propagation completion flag; orbit correction flag; initial conditions file name; orbit correction schedule file name; file with a-priori state vector covariance matrix; the name of the file with Chebyshev's approximating polynomial coefficients.

The *Executive Module* uses information from the *Service Database* as the input to propagate the orbit of the satellite attributed to the given slot. The executive module integrates equations and calls the *Ephemeris Generator*. This object approximates coordinates of the satellite by Chebyshev's polynomials. Polynomial coefficients are stored in the *Service Database* in a unique file for each slot.

The Tracking Stations Scheduler Tool

The Tracking Stations Scheduler, or **TSS** program is designed to plan operation of tracking stations. The objective is to provide a required level of orbit determination accuracy by minimization of ground tracking station operation time taking into account technical restrictions, such as observation conditions, antenna <<switching time>>, etc. The tool computes a posterior covariance matrix for each satellite in absolute Cartesian or orbital frame and total operational time to provide the required accuracy of orbit determination. The tool also allows scheduling the network of tracking stations to work with several constellations simultaneously.

The TSS tool employs discrete Kalman's filtering algorithm and Pontryagin's necessary conditions of optimality. Linear model of orbital motion obtained from non-linear equations by linearization in vicinity of reference motion in central gravity field is used. Initial conditions for linearization can be obtained either using the CONSIM tool, or the orbit determination procedure. The model of tracking station motion considers precession, nutation, sidereal time and instant, the Earth's pole displacement. The measurement channel model considers range and range-rate measurements taking into account systematic and random errors.

The tool retrieves input data from the Global Database. Local database is assigned to Store files of the constellation satellite ephemeris, initial conditions and a posterior covariance matrix of satellites.

Schedule table stored in database contains schedules for each tracking station working with specified constellation. Names of tracking stations (TS); a constellation; the file with initial data for scheduling; the file with observation zones of the TS; the file with schedules of a given TS are stored in the schedule table.

The Constellation Control Tool

The Constellation Control, or **CONCONTR** tool computes constellation control strategy (long term planning) taking into account distortion due to the Earth's oblateness, atmosphere drag and solar radiation. This program can be used for R&D purposes (a priori analysis), as well as for decision making on either partially or completely deployed constellation control in a long run. At the R&D phase, the tool can be used to specify required propellant load, orbit correction rocket engine parameters, etc.

This tool is useful for examining design definition of a satellite system at the R&D phases as well as to make decisions about constellation corrections during the system exploitation. Input data to the tool are:

- *constellation geometry:* an orbit altitude; inclination; number of orbits; number of satellites in each of the orbits;
- *Satellite performances:* propellant amount (characteristic velocity); mass, aerodynamic reference area; drag force coefficient; reflection coefficient;
- *orbit correction engine characteristics:* minimum and maximum characteristic velocity impulses; thrust error characteristics;
- *observation instrument performances:* view angle; spectral band; resolution;
- *observation project;*
- *observation requirements:* minimum number of observation tasks to be solved; maximum allowed time delay of satellite passage over the specified region with respect to nominal passage moment;
- *environmental data:* parameters of the Earth's gravity field, atmosphere drag and solar radiation pressure models.

Constellation keeping strategy is figured out considering distortion due to the Earth' s gravity field, aerodynamic drag and solar radiation pressure. Orbit correction errors are taken into account as well. A simplified atmosphere model is used. It does not consider density variation with respect to either time of a day, or a solar activity. Maximum of density is used to provide a safe result. Solar radiation pressure is assumed as constant; it is evaluated for a specified aerodynamic reference area of the satellite. Since all the satellites are in similar orbits, relative gravitation effects are not considered because all the satellites are disturbed in a similar manner. Random orbit correction errors are also considered.

Output of the constellation maintenance strategy are intervals between constellation corrections, correction impulse magnitudes, propellant exhaust expected time.

Orbit correction instants are computed when the condition of user requirements are violated. The constraints on the constellation effectiveness (number of observation tasks to be solved), or on delay of a passage over a particular ground site are taken into account. The user is allowed to vary requirements to make "what-if" experiments. Time and number of corrections is estimated followed by moments of a propellant exhaust evaluation.

The Orbit Control Tool

The Orbit Corrections Scheduler, or **ORCOS** program, compiles correction sequence to transfer a satellite into a specified position within its constellation. Computations are based on orbit prediction generated by the CONSIM tool. ORCOS checks for orbit parameters of satellites and generates orbit correction program if the parameters are not in the control box. ORCOS computes the schedule in the CONSIM compatible file format, and makes changes to the constellation database. The user is allowed to set correction start date and time, specify optimization criterion (time or propellant), inclination correction mode and constraints common for all the slots. Individual control box for each orbit should be defined in Global Database. Only near circular orbits are considered.

It is assumed that the satellite is equipped with three-axis attitude control system and thrusters producing control accelerations. The attitude control system makes the satellite oriented such that transversal and normal to orbital plane thrust direction remains constant during a powered phase.

Transversal acceleration controls the argument of latitude, drift and eccentricity (in-plane correction, or slot position correction); whereas normal acceleration is used to control inclination.

Slot position corrections and inclination corrections never go simultaneously. Moreover, the inclination correction program follows slot position correction program.

A thruster operates in on-off mode, so the orbit correction procedure consists of coast and powered phases. The coast phase duration should be not less than a specified value. No thrust errors are considered.

The objective is to derive the number of corrections (powered phases), and thrusters on-off schedule which ensures required terminal conditions.

The working algorithm implemented in the program is constructed using several partial analytical solutions combined with numerical exhaustive search. Since slot position and orbit inclination corrections are assumed as separated, it is possible to consider independent sets of equations. Partial analytical solutions are: one-powered-phase angular position correction algorithm, or A-1 algorithm; two powered-phases time-optimal angular position correction algorithm, or A-2 algorithm; numerical terminal control algorithm, or N-2 algorithm; optimal, with respect to minimum of characteristic velocity, analytical algorithm for orbit inclination correction, or A-IC algorithm.

All the algorithms are built with assumption that a satellite remains in vicinity of circular reference orbit. Perturbing factors such as the Earth's oblateness, atmosphere drag, and others are not considered.

The Multi-Satellite Orbit Determination Tool

The Multi-Satellite Orbit Determination, or **MSOD** tool computes statistics of the constellation expanded state vector and satellites phasing deviations as well as variances with respect to nominal values set by the constellation geometry.

The statistics are estimated using linear model. Nonlinear source equations are linearized in a vicinity of actual state vector obtained from orbit determination for a single satellite, Deviations are computed using results of a single satellite orbit determination and nominal parameters.

This tool also uses the local Database, which stores initial conditions and ephemeris of a constellation of interest.

The executive module contains: generator of linearized relations; constellation analyzer; output data generator.

Generator of linearized relations computes partial derivatives with respect to the components of the state vector in absolute Cartesian frame and with respect to osculating elements. All partial derivatives are computed from actual satellite state vector obtained from a single satellite orbit determination.

Constellation analyzer computes actual values of osculating elements using corresponding nonlinear formulae as well as standard deviation of osculating elements using linearization. Deviations from reference values are also computed.

The Mission Planning Tool

The Mission Planning, or **MISPLAN** tool is designed to form timelines for the Earth observation satellite constellation consisting of one or several CBERS type satellites. Simultaneously timelines for ground imagery data acquiring stations are formed.

The timelines for on-board instruments CCD camera, Infra Read Multi Spectral Scanner (IR MSS) and Wide Field Imager (WFI) are formed separately considering customer requests for imagery, constellation status and technical constraints.

Imagery that is provided by IR MSS and WFI can be downloaded to ground stations in *real time* mode. Therefore, image downloading sessions for those instruments are planned for each pass of the satellite over the ground station with the only exception: local nighttime for WFI instrument. Imaging program is *on-off* type schedule, consisting of imaging sessions during satellite passages over the ground stations. Manual editing of sessions by the operator for both the IR MSS and WFI instruments is enabled.

The CCD camera is scheduled for customer requests on specified ground targets arriving to the Mission Control Center. Request include information about geographical coordinates of a ground target; spectral bands; temporal constraints; priority of the request; cost of the image.

That instrument can operate either in *real time, or tape replay* modes of data downloading. Therefore, the control to figure out involves schedules for camera, tape recorder, and radio link. Three operation modes for on-board equipment are considered;

- real time imaging of ground targets with reorientation of camera field of view and data downloading to a ground station;
- imaging of targets outside of a ground station's visibility zones and data recording on the onboard tape recorder;
- delayed transmitting of recorded data during satellite passes within ground station visibility zone.

A timeline optimization procedure for CCD camera consists of the following steps:

- orbits propagation;
- determination of possible snapshots of requested targets and data downloading sessions;
- WFI camera programming;
- IR MSS camera programming;
- CCD camera programming.

The first step is performed using the CONSIM tool. Sets of possible snapshots for a given list of ground targets and down link sessions between satellites and ground stations are formed at the second step.

Filtering of possible download sessions is performed to form timelines for WFI and IR MSS instruments, considering lightening conditions. An operator can make manual editing of both timelines. The total mean expected income resulted from the timeline execution is utilized as a timeline optimality criterion for the CCD camera. Two optimization algorithms are implemented in the tool: *sequential assignment* algorithm and *sorting out* algorithm. The first one provides choosing the best snapshots for including into timeline in presence of user requirements and technical constraints in sequence, starting from requests with major priority, maximum cost of image and minimum side view angle. Decisions about targets and snapshots, included into the timeline, made at previous steps of that sequential process, are not changing, when snapshots of next requests from the list are considered for inclusion into the timeline.

Choice of optimal snapshot sequence in the *sorting out* algorithm is performed by forming and evaluating all possible combinations of snapshots, starting from combinations, consisting of targets, that are major priority, maximum cost and minimum side view angle.

The Operative Mission Performance Analysis Tool

The Mission Analysis, or MISSAN tool computes a set of indexes in addition to the major timeline optimality criterion used in the TMISPLAN tool. The indexes characterize performances of the request list fulfillment and loading of different components of the satellite system during the period of interest.

Performances are estimated with respect to:

- total list of requests;
- different priority groups of requests;
- different periods of time.

Satellite system components loading is estimated with respect to ground stations and satellite on-board instruments:

- pointing angle variations;
- tape recorder capacity usage;
- downlink sessions usage;
- imagery data delivery time.

The performances are estimated both for the constellation as a whole and for separate satellites. Mean value, standard deviation, and histogram of absolute and relative number of requests included into the timeline are estimated for the CCD camera.

For WFI and IR MSS estimated are the statistics, characterizing total time within a planning period when these instruments are in use.

An example of mission operative performances is shown in the chart, presented in Fig. 2, demonstrating impact of satellite number in a constellation upon a number of targets (from the total 96 requested targets at the Brazilian territory) included into optimal timelines for CCD camera of CBERS satellites versus the timeline duration.

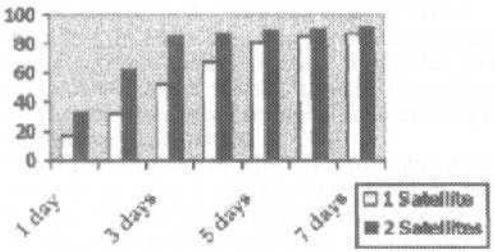


Fig. 2 Number of targets included into timelines for CBERS 1 and 2 satellites versus timeline duration

The Flight Operations Plan Compiler Tool

The Flight Operations Plan Compiler, or **FOPC** tool is designed for routine operations computer-assisted automation in the Mission Control Center. FOPC compiles the Flight Operation Plan Pages automatically. The operator is allowed to edit the compiled plan. The main list of FOPC functions is:

- Flight Plan editing;
- station passage predictions,
- antenna pointing computation,
- total contact time determination,
- flight control procedures, sequences and tele-command database management.

The FOPC tool works with command database. Each command has a code and parameters. It is possible to pack a set of commands in a sequence identified by a unique name. The program allows to create and to edit the Flight Control Plan consisting of commands. The user selects the constellation and the satellite for which the Flight Plan has to be compiled. At first, all the commands with priority flag set are inserted. Then the user picks up commands to append manually. For a given orbit number the program computes visibility zones; entry/exit points are figured out. Azimuth and elevation of an antenna is computed for entry and exit taking into account minimum elevation condition. In case of overlap of visibility zones of several stations the total contact time is computed.

Satellite track for a given orbit, visibility zones are displayed as can be seen on the World map from space. Station entry/exit time is shown in a separate window for each ground station. Information on the last shadow passage and time when the antenna will be pointed either to zenith or to the Sun is also available. In case there is a contact with the satellite of interest, the Flight Control Plan is enabled for editing.

The Flight Control Plan contains tracking stations calibration session begin time, and the telemetry expected status. The user is allowed to pick up commands from the database, add or delete commands and command sequences, to pick up ground station, edit telemetry status. All the plans compiled are stored in the database and can be printed.

Conclusion

The Space System Toolbox possesses various tools useful for solution of different problems relaying on LEO satellite constellation creation, control and management. Major application areas of the tools are:

- determination of optimal parameters of the constellation based on a priori mission performance analysis;
- creation of the constellation deployment and control strategies;
- constellation satellite orbits prediction and control;
- optimal scheduling of tracking stations resources for navigational purposes;
- optimal timeline forming and operational mission performance analysis for the CBERS type satellite constellations;
- computer-assisted designing of routine operations for satellites using data provided by other operational tools and global database.

Preparatory studies have been started on definition and development of specified tools for R&D, control, mission planning and performance analysis of LEO communication satellite constellations, as well as software for real time satellite simulation.

RBCM - J. of the Braz. Soc. Mechanical Sciences ISSN 0100-7386
Vol. XXI – Special Issue - 1999 - pp. 57-63 Printed in Brazil

Sharing the use of a Satellite Under Quota Constraints : an Overview of Methods

Eric Bensana
Michel Lemaitre
Gérard Verfaillie
Office National d'Etudes et de Recherches Aérospatiales (ONERA - CERT)
2, avenue Edouard Belin, BP 4025, 31055 Toulouse Cedex 4, France
(eric.bensana, michel.lemaitre, gerard.verfaillie) @cert.fr

Nicolas Bataille
Centre National d'Etudes Spatiales (CNES)
18, avenue Edouard Belin, 31401 Toulouse Cedex 4, France
nicolas.bataille@cnes.fr

Abstract

The purpose of this paper is to present different methods to deal with a decision problem characterized by the exploitation of an earth observation satellite which must be shared between the agents which co-funded it. The problem is to decide on the daily selection of a subset of pictures, among a set of candidate pictures which could be taken the next day considering the satellite trajectory. This subset must satisfy three kinds of constraints : physical problem (hard) constraints; efficiency constraints, aiming at maximizing the satisfaction of each agent; and a fairness constraint, which is ideally satisfied when each agent receives an amount of the resource exactly proportional to its financial contribution. Efficiency and fairness constraints are usually antagonistic. Although fair division problems have received considerable attention for a long time, especially from microeconomists, this specific problem does not fall entirely within a classical approach, because the candidate pictures may be incompatible and also because a picture is only of value to the agent requesting it. We investigate and propose different ways for solving this share problem : regulation, mono-objective optimization with priority to fairness or to efficiency, multi-criteria optimization.
Keywords *: multi-criteria decision, optimization, multi-agent resource management*

Context of the studies

Because of their cost, space projects such as earth observation satellites, space stations or space probes are often co-funded by several agents (countries, companies, entities ...). Once operational, the common property resource must be exploited and shared in a way which satisfies three kinds of constraints :

- *physical* constraints : they impose limits for the exploitation of the resource; for example no more than *m* pictures can be taken at once, provided there are only *m* instruments on board; a minimal transition time between two pictures taken by the same instrument must be respected; on board memory and power is limited ...
- *efficiency* constraints : each agent wants to get the highest possible satisfaction in return;
- a *fairness* constraint : each agent must get a return on investments proportional to its financial contribution to the project; the better the proportionality of returns is achieved, the more the share quality improves.

The first kind of constraints must absolutely be met (hard constraints) whereas the two others are usually preference constraints (soft constraints). As it can be easily guessed, the efficiency and fairness constraints are antagonistic: the search for a perfect share leads to poorly efficient decisions, and conversely, decisions which maximize the global satisfaction of agents are often unfair. So, a compromise between the best satisfaction of both constraints must be found.

The case involving only one agent (in which case there is no fairness constraint) is usually a difficult combinatorial discrete optimization problem and is a perfectly well stated problem. The multi-agent case is also a discrete combinatorial problem, but is actually a multi-objective optimization problem (R. L. Keeney et al., 1976); the first difficulty arises when searching for a meaningful and principled definition of a good compromise between efficiency and fairness.

This article sums up a set of studies, the aim of which was to improve the understanding of this type of problem and to propose methods to solve a specific share problem, namely the fair and efficient exploitation of an earth observation satellite owned in common by several agents. The next section sets the problem more formally and then different approaches and methods devoted to the resolution of this share problem are presented.

Multi-criteria decision problem formulation

To facilitate the understanding, the context will be relative to the exploitation of an Earth observation satellite, co-funded by several agents but the concepts developed remain valid in the general case.

These agents make daily requests for pictures they would like to be taken by the satellite. Roughly speaking, the problem consists in selecting each day, among the set of candidate pictures which could be taken the next day considering the satellite trajectory, a subset of pictures which satisfies all the physical constraints, maximizes the satisfaction of the agents, and respects as far as possible a fairness constraint. Such a selection will be called a *decision*. The satisfaction and fairness constraints will be taken into account over a fixed interval of several days called *history*.

Let us describe more formally the problem. First, the data:

- there are n agents;
- D_{ik} is the set of pictures requested on day k by the agent i ; the sets D_{ik} are disjointed and D_k is the union of the D_{ik} for all the agents.
- each picture in D_k could be taken the day k, but all pictures cannot be taken because the physical hard constraints must be satisfied; a subset $X \subseteq D_k$ is said *admissible* if all pictures in X satisfy the hard constraints and hence can all be taken on day k;
- $w(x)$ is the weight of the picture X; it is freely set by the agent requesting the picture, and reflects its importance for the agent;
- $c(x)$ is the *real cost* of the picture; the cost is supposed to be independent of the agent requesting the picture, and to have been fixed by mutual agreement between agents; costs can be based for example on physical consumptions on board during the shot for the picture : time, energy, memory ...
- q_i is the quota of agent i and is proportional to its financial investment ($\sum_{i=1..n} q_i = 1$)

Each day k-1, the demands D_{ik}, with corresponding weights, are collected and the selection algorithm must compute the sets of pictures A_{ik} ($A_{ik} \subseteq D_{ik}$) which will be shot for the agent i the day k. These A_{ik} are such that:

- the union A_k of the A_{ik} for all the agents must be *admissible* (physical hard constraints);
- the *cumulative satisfaction* of each agent must be as high as possible (efficiency constraints) : the *satisfaction* of the agent i the day k is measured by $s(A_{ik})$ where $s(X) = \sum_{x \in X} w(x)$; the cumulative satisfaction, measured over a given interval of days H ending on the day k over H for the agent i is then $cs_i = \sum_{k \in H} s(A_{ik})$
- the quality of the share over H must be as high as possible (fairness constraint) : the *use* of the resource by an agent i on day k is measured by $c(A_{ik})$ where $c(X) = \sum_{x \in X} c(x)$;; the *cumulative cost* a given interval of days H ending on the day k is then $cc_i = \sum_{k \in H} c(A_{ik})$; the real relative use r_i of the satellite by agent i is defined by $r_i = cc_i / (\sum_{j=1..n} cc_j)$

The problem above is stated as a sequence of multi-objective optimization problem instances. However, the fairness constraint is not yet formally stated.

We have investigated different methods devoted to the resolution of this share problem (that is general schemes for computing the A_{ik}). Each one is based on a particular way of taking into account the fairness constraint and the necessary compromise with the efficiency constraints.. The first method considers the decision algorithm as a process which can be controlled by the mean of the weights assigned to the pictures, the second and third classes of methods reduce the problem to a sequence of mono-objective optimization problem instances, whereas the last one focuses on the multi-objective aspect.

Different approaches experimented

Different approaches have been studied in the last two years to deal with such decision problems. The purpose of the paper is to present the main characteristics of the methods developped. All have been implemented and tested in simulation in the context of a shared earth observation satellite.

Regulation approach

This approach was the first one studied, it is based on *control theory* models. The process to control is the selection algorithm; the weights used to compute the satisfaction of the agents are the control inputs and the outputs are for each agents the measured ratio of use of the resource and a regulation loop is built upon the daily selection algorithm.

The aim is to get over time a behaviour of the system such that when an agent got more than its quota the weights of the pictures requested by him must be decreased and when he got less than its quota the

weights of its pictures must be increased. The fairness constraint is not maximized each day but regulated.

Both standard PID and fuzzy controllers (A. Rachid, 1996) have been designed, but to simplify, only PID related aspects will be presented. The command delivered by a PID controller at time k is computed by (discrete case) :

$$u_k = u_{k-1} + P.(\varepsilon_k - \varepsilon_{k-1}) + I.T.\varepsilon_k + D/T(\varepsilon_k - 2.\varepsilon_{k-1} + \varepsilon_{k-2}) \quad (1)$$

where ε_k is the difference between the command and measured output at time k; P, I and D are the parameters of the controller for the proportional, integral and derivative part; T is the sampling period.

In our context, $T=1$ and the controller computes one command u_{ik} for each agent i, with equation 1. The ε_{ik} are the difference between the required quota and the measured relative use of the satellite by agent i :

$$\varepsilon_{ik} = q_i - r_i \quad (2)$$

As the u_{ik} can be positive or negative, a correction function f is used to map the u_{ik} into the interval $[0,+\infty[$ and to compute the α_{ik}, the only constraint for the choice of f being that when $u_{ik} = 0$ then α_{ik} must be 1.

Functions like $f_a = (1 + arctg(u)/\pi)^M$ or $f_e = e^{min(u, M)}$ can be used. The M parameter allows to set limit for the values for the α_{ik} : $0 \le \alpha_{ik} \le 2^M$ for f_a or $0 \le \alpha_{ik} \le e^M$ for f_e.

Then for each agent i and each picture $x \in D_{ik}$, the weight $w'(k,x) = \alpha_{ik} \cdot w(x)$ is computed. These weights are those used by the selection algorithm to compute the set of pictures :

$$A_k = argmax\{s'(X) \mid X \subseteq D_k \text{ and } X \text{ is admissible}\} \quad (3)$$

with $s'(A_k) = \Sigma_{i=1..n}\, s'(A_{ik}) = \Sigma_{i=1..n}\, \alpha_{ik} \cdot s(A_{ik})$

It should be noted that when the correction function f is the identity function, pictures can get negative weights, and thus cannot be selected by the algorithm to belong to A_k. This option is equivalent to *a priori* limiting the set of candidate pictures D_{ik} by filtering them.

Results

The regulation loop has been implemented on top of a selection algorithm based on dynamic programming with different parameters to specify the type of controller to use (PID or fuzzy, correction function f, value of M ...).

Several successfull controllers have been tested. Results show that the approach works well. The fairness constraint is satisfied at the end of the first week of simulation (simulations were run on 70 days) and the difference between the measured and expected quota is less than 1%. Best results are obtained with controllers of type PI (the D coefficient is set to 0).

Advantages of this approach are its simplicity and the fact that the method is independent of the selection algorithm since only the weights of the pictures need to be adjusted; so it can be easily implemented in an existing application. The main difficulty lies in the necessary adjustment of the parameters of the controller.

Fairness first methods

This class of methods emphasizes the fairness constraint and tend to improve its satisfaction at efficiency expenses. Several variants have been designed based on a filtering of the demand or a preallocation for each agent of time windows where it has the full property of the resource.

Limitation of the demand

A first method consists in filtering the demand in order that the quotas are respected in the set D_k of the required photographs which will be considered for the scheduling phase. Then the demands are scheduled independently of the fairness constraint. This method lead to rather acceptable results but was not considered worth to investigate deeper.

As limiting *a posteriori* the demand may seem strange, this approach can still be interesting if we imagine that each day, before the agents set their demand, we compute for each of them the maximal amount of use of the resource he is allowed. Each agent must then make his own filtering before submitting the D_{ik} to the selection algorithm.

A priori sharing of the resource

The entitlement to use the resource is shared by allocating observation windows to each agent in turn. Observation windows are merely sequences of successive orbits of the satellite. Each day, the agent i is given the right to freely exploit about $q_i.n$ orbits, where n is the number of orbits daily covered by the satellite. Observation windows are assigned to agents on the basis of a fixed repetitive procedure. This procedure and the trajectory of the satellite are such that each agent gets opportunity to shoot any place in the world within a bounded number of days.

Following this method, the whole problem can be cast into a set of optimization problem instances, one for each agent each day.

Assuming that each $x \in D_{ik}$ belongs to the window assigned to agent i the day k, the successive optimization problem instances consist in maximizing the satisfaction of agents by finding :

$$A_{ik} = argmax\{s(X) \mid X \subseteq D_{ik} \text{ and } X \text{ is admissible}\} \quad (4)$$

This problem can be seen as a discrete constrainted optimization problem and we rely on a specific branch and bound algorithm (G. Verfaillie et al.) within the valued CSP framework (T. Schiex et al., 1995) to compute to almost all windows the optimum.

These simulations show for this method a very good quality share: the number of pictures effectively selected and assigned to each agent is very close to a number proportional to its quota. But the decisions are clearly inefficient, when compared with those resulting from other methods.

A possible refinement for this method is to allow, once an agent i has booked all its pictures for a given window, to allow other agents to try to book some of their pictures when agent i do not use the resource. By doing so efficiency can be improved, but fairness, of course, is no more fully satisfied.

Results

These methods usually provide a quite perfect share, but a poor satisfaction level. Moreover this approach becomes more complex if other resources than time has to be *a priori* shared. For example if the memory limitation must be taken into account on the whole day, then a share of memory must be assigned also for each agent.

Efficiency first

This class of methods considers the opposite view: priority to efficiency and fairness if possible. It is based on three main ideas:

1. For efficiency, maximize each day a linear combination of individual satisfactions of the agents;
2. for fairness, choose this combination in a way favoring the fairness constraint;
3. check that each agent has obtained a *fair share*.

The last point is borrowed from the literature on fair division (H. P. Young, 1994, H. Moulin, 1995, S. J. Brams et al., 1996) : a decision is fair when each agent receives at least a *minimal fair share*, defined for the agent i as q_i times the satisfaction it would get if it were the only user of the resource. More formally, the fairness constraint is considered to be satisfied if for each agent i :

$$cs_i \geq q_i \, . \, cs_i^M \quad (5)$$

with $cs_i^M = \Sigma_{k \in 1l} \, s^M(D_{ik})$ and

$$s^M(D_{ik}) = max\{s(X) \mid X \subseteq D_{ik}, \text{ and } X \text{ admissible}\}$$

The weights of pictures are considered as monetary bids. As weights are freely fixed by the agents, we must make satisfactions comparable by *normalizing* them. The function to be maximized is then :

$$s'(A_k) = \Sigma_{i=1..n} \, s'(A_{ik}) = \Sigma_{i=1..n} \, \alpha_{ik} \, . \, s(A_{ik}) \quad (6)$$

where the coefficients α_{ik} have to be determined. It can be easily demonstrated that the value of the α_{ik} must be $\alpha_{ik} = q_i / s^M(D_{ik})$. The set of daily selected pictures with this method maximizes under admissibility constraints, the function

$$s'(A_k) = \Sigma_{i=1..n} \, [q_i \, . \, (s(A_{ik}) / s^M(D_{ik}))] \quad (7)$$

With this choice for the coefficients α_{ik}, the selected decisions are independent of the scale of weights used by each. However, the method does not guarantee the satisfaction of the fairness constraint, it will have to be checked *a posteriori*. Hopefully, it has a lot of chance to be satisfied, for two reasons :

- a structural reason : the normalization of the weights tends to favor agents with upper quotas, in a direction favorable to the satisfaction of the fairness constraint; moreover, the fairness constraint is rather soft;
- a statistical reason : when there is a large number of candidate pictures, not too tightly incompatible, the structural reason can exert its influence; this is the case with our (realistic) simulation data: the simulations show that the fairness constraint is always widely satisfied

A variant of this method has been designed, for the case where the fairness constraint could not be satisfied, *i.e.* when requests are poorly distributed geographically and highly incompatible. This variant is inspired by the classical Knaster's procedure of sealed bids. Each day virtual monetary compensations between agents are computed, reflecting the gap between the actual and ideal shares. An agent having a positive credit is ``late" on its quota (it received not enough pictures selected) and conversely, an agent with a negative credit is ``ahead" on its quota. These compensations are used to modify the above normalization procedure for the next days in a direction favorable to a fairest share. As the regulation approach is very close to this approach, it can be seen also as a kind of efficiency first approach.

As the maximized function is a linear combination of individual satisfactions of the agents, decisions selected by this method are Pareto-optimal decisions. A Pareto-optimal decision is a non-dominated decision in the n-dimensional space of individual satisfactions[1].

Such decisions are also called *efficient* decisions because it is impossible to improve a decision selected by this method for one agent without reducing the satisfaction of at least another agent. This property explains the good satisfaction levels obtained with this method in our simulations and justifies the name ``efficiency first".

Results

This method and its variant have been implemented successfully using the same Valued CSP framework as before. However, the number of instances to be solved is large (all the $s^M(D_{ik})$ must be computed) and the size of the whole instance (for the maximization of $s'(A_k)$) may be very important.

As expected this method gives the best satisfaction, but a price in quality of share must be paid for it, which is quite acceptable, since the fairness constraint is in general widely satisfied (each agent gets more than his minimal fair share).

Multicriteria approaches

These approaches tend to compute a set of good compromise decisions instead of giving priority to fairness or efficiency. The most precise way to set the whole problem is to formulate it as a sequence of multi-criteria discrete optimization problems. The criteria to be maximized would be:

- the n agent's satisfaction criteria cs_i for $i=1...n$;
- a criterion j measuring the quality of share, which has to be defined.

Only the set of Pareto-optimal decisions in this $n+1$ dimensional space are worth considering. The approach which consists in collecting this set of decisions is unworkable, because it is very large (in our application).

Fairness as a hard constraint

The first idea is to select the fairest decision within the set of efficient decisions. This can be done by taking into account the fairness as a hard constraint during the selection.

In practice, the respect of the share quotas can be costly in term of satisfaction of the entities : without these quotas, the overall satisfaction of the entities would be much better (but of course not well shared). It would be very costly to try to respect too strictly the quotas. That is why, in practice, a compromise has to be done between the quality of respect of the quotas and the satisfaction of the entities. For example, a difference of 10 % with the required quotas could be accepted if the global satisfaction of the entities would be sufficiently improved.

Two variants have been experimented :

- search among the solutions which optimizes the global satisfaction of the agents, the solution which is the nearest from the required quotas.

---

[1] Given a set of criteria, a decision D dominates another decision D' for these criteria if and only if D is better or equal than D' for all criteria, with at least one criteria for which D is strictly better than D'.

- searches, among the solutions which are nearly optimal for the satisfaction criterion (for example ≥ 95 % of the optimal value), the solution which is the nearest from the required quotas.

Although giving good results on small examples, this approach is in fact too much computation demanding to be used on real size problems.

A more pragmatic multicriteria approach

So, we have to resign ourselves to aggregate some criteria. A sensible solution is to aggregate individual satisfactions into a global cumulative satisfaction *gcs*, and to keep apart the quality of share criterion *j*. Eventually, potentially interesting decisions can be presented in the two-dimensional space *j*×*gcs*. The aim is to help a human decision-maker to make decisions, by providing him with interesting compromises.

As in the regulation approach, the share is measured upon some function of the *real cost* of pictures, such as time, memory or power consumption on board. By doing so, we are sure to remain insensitive of the choice of the weights which are agent-dependent.

The *quality of share over H* is measured by a "distance" between the real use of the satellite r_i and the quotas q_i. *Inequality indices* (S. J. Brams, 1988), like the Gini indice, developed by microeconomists can be used to base the function *j* measuring the quality of share :

$$j = 1 - \tfrac{1}{2} \left(\sum_{1 \leq i,j \leq n} |r_i \cdot q_j - r_j \cdot q_i| \right) \qquad (8)$$

j is in the range 0 to 1 and $j=1$ when the share is perfect *i.e.* costs of obtained pictures exactly proportional to quotas.

The *global cumulative satisfaction* of agents over the interval *H* is measured by a linear combination of normalized cumulative individual satisfactions :

$$gcs(A_k) = (1/n) \sum_{i=1..n} (cs_i / cs_i^M) \qquad (9)$$

gcs is in the range 0 to 1 and the maximum 1 is reached when each agent is satisfied as much as it can be if it were the only owner of the resource; *gcs* is independent of the individual scales of weights and of the quotas.

Results

This method stay very costly in term of computational resource. The set of Pareto-optimal decisions in the *j*×*gcs* space can be computed exactly by a branch-and-bound search, or approached by an adapted local search method when the search space is too large. To limit the number of decisions presented, we can limit to a subspace of the *j*×*gcs* plane by imposing that criteria must be greater than a given limit (for example Pareto-optimal decision with $gcs \leq 0.5$ are not interesting).

Results of simulation confirmed that this method gives compromise solutions between the ones provided by fairness first or efficiency first methods.

Experimentation

The data used for these experimentations comes mainly from scenarios established to study the problem of scheduling the photographs of future earth observation satellites.

It is impossible, in this short article to provide extensive and detailed results for all the methods. We will restrict ourselves to present the different contexts used for the simulations.

First context

The regulation approach was simulated in a multi-agent satellite context. There are 3 agents with quotas (0.07, 0.465, 0.465) and globally between 500 and 700 daily candidates pictures. The weights are in the range 0 to 1, but are fixed independantly of the agents. The cost of a picture is based on the time of use of the satellite and differs from one picture to another. Simulations were run over a period of 70 days.

Second context

Other methods have been simulated in a context extrapolated from the simulated demand concerning the future Spot5 satellite (G. Verfaillie et al., 1996, E. Bensana et al., M. Lemaître et al., 1997), which the assumption of three cameras on board. This data, provided originally for the mono-agent case, has been adapted to simulate a demand from $n=3$ agents. Simulated agents request each day about the same number of pictures. The quotas for the simulations are (0.1, 0.3, 0.6). Weights are in the range 1 to 100.

We dispose of data for 371 days. The most loaded day comprises 427 requested pictures. The cost function is simply $\forall x,\ c(x)=1$ (that is, we only count the number of selected pictures). The interval of days H on which cumulative satisfaction and cost functions are based is always the whole history $H=[1..k]$, where k is the present day.

Conclusion

We have described a specific share decision problem involving multiple agents, in which the satisfaction of two kinds of constraints poses a dilemma: efficiency constraints aim at satisfying the agents the most, whereas a fairness constraint watches over equity among agents. We proposed different approaches and methods to solve this problem.

The first method was to consider the daily selection algorithm as a process whose control inputs are the weights of the pictures submitted.

The second class of methods gives priority to fairness first, and then for efficiency. They are simple *a priori* sharing method, modyfing the demand in order to respect the quotas at the demand level or allocating observation windows to each agent in turn. It results in very good shares, but inefficient decisions.

The third method gives priority to efficiency and satisfy fairness if possible. A global satisfaction criterion is defined and maximized and a *minimal fair share* for each agent is defined *a priori* but only checked *a posteriori*. It delivers quite good decisions where minimal fair shares are always achieved and the global satisfaction is high, and uses a tolerable amount of computational resources.

The last approaches do not favor one constraint or the other, but compute a set of good compromise decisions. This is a multi-criteria approach, based on the computation of a subset of Pareto-optimal decisions in a two-dimensional space : global satisfaction of all the agents, quality of the share. It is very costly in computational resources, but allows a human decision-maker to preview a set of interesting non-dominated compromise decisions.

The overall conclusions of this work are:

- no method can be indisputably put forward; the problem is not to choose a method against another one, it is to present to the agents a set of methods and their properties and to let them decide according to the properties they consider the most important;
- whereas general methods of sharing can be stated, each share problem is specific and must be studied carefully;
- discrete share problems like this one are computationally very consuming; more specialized combinatorial optimization algorithms are needed to solve them.

References

R. L. Keeney, and H. Raiffa, 1976, "Decisions with Multiple Objectives : Preferences and Value Tradeoffs." John Wiley and Sons.

A. Rachid, Editor, 1996, "Systèmes de régulation." Masson.

G. Verfaillie, M. Lemaître, and T. Schiex, "Russian Doll Search for Solving Constraint Optimization Problems." AAAI 96. ftp://ftp.cert.fr/pub/verfaillie/rds-aaai96.ps

T. Schiex, H. fargier, and G. Verfaillie, August 1995, "Valued Constraint Satisfaction Problems Hard and Easy problems." In proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, pages 631-637, Montreal, Quebec, Canada, August 1995. ftp://ftp.cert.fr/pub/verfaillie/ijcai95.ps

H. P. Young, 1994, "Equity in Theory and Practice." Princeton University Press.

H. Moulin, Cooperative Microeconomics, 1996 "A Game- Theoretic Introduction." Prentice Hall, 1995.

S. J. Brams, and A. D. Taylor, 1996, "Fair division : From cake-cutting to dispute resolution." Cambridge University Press. 1996.

H. Moulin, 1998, "Axioms of Cooperative Decision Making." Cambridge University Press.

G. Verfaillie, and M. Lemaître, "Optimisation de la programmation et partage d'un satellite d'observation." Technical report of the CNES R&D study "Intersolve", ref CERT 2/3571/DERI.

E. Bensana, G. Verfaillie, J.C. Agnèse, N. Bataille, and D. Blumstein. "Exact and Approximate Methods for the Daily Management of an Earth Observation Satellite." SpaceOps 96, Munich, Germany. ftp://ftp.cert.fr/pub/verfaillie/spaceops96.ps

M. Lemaître, G. Verfaillie., July 1997, "Daily management of an earth observation satellite : comparison of ILOG Solver with dedicated algorithms for Valued Constraint Satisfaction Problems." Proceedings of the Third ILOG International Users Meeting, Paris, France. ftp://ftp.cert.fr/pub/lemaitre/Papers/97-ILOG.ps

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 64-71

ISSN 0100-7386
Printed in Brazil

Message Mode Operations for Spacecraft: A Proposal for Operating Spacecraft During Cruise and Mitigating the Network Loading Crunch

Ed Greenberg
Merv MacMedan
Greg Kazz
Pieter Kallemeyn
Jet Propulsion Laboratory, California Institute of Technology
4800 Oak Grove Dr., Pasadena, CA 91109 USA
greg.j.kazz@jpl.nasa.gov

Abstract

The NASA Deep Space Network (DSN) is a world-class spacecraft tracking facility with stations located in Spain, Australia and USA, servicing Deep Space Missions of many space agencies. The current system of scheduling spacecraft during cruise for multiple 8 hour tracking sessions per week currently leads to an overcommitted DSN. Studies indicate that future projected mission demands upon the Network will only make the loading problem worse. Therefore, a more efficient scheduling of DSN resources is necessary in order to support the additional network loading envisioned in the next few years: The number of missions is projected to increase from 25 in 1998 to 34 by 2001. In fact, given the challenge of the NASA administrator, Dan Goldin, of launching 12 spacecraft per year, the DSN would be tracking approximately 90 spacecraft by 2010. Currently a large amount of antenna time and network resources are subscribed by a project in order to have their mission supported during the cruise phase. The recently completed Mars Pathfinder mission was tracked 3 times a week (8 hours/day) during the majority of its cruise to Mars. This paper proposes an innovative approach called Message Mode Operations (MMO) for mitigating the Network loading problem while continuing to meet the tracking, reporting, time management, and scheduling requirements of these missions during Cruise while occupying very short tracking times. MMO satisfies these requirements by providing the following services:

- *Spacecraft Health and Welfare Monitoring Service*
- *Command Delivery Service*
- *Adaptive Spacecraft Scheduling Service*
- *Orbit Determination Service*
- *Time Calibration Service*

Utilizing more efficient engineering telemetry summarization and filtering techniques on-board the spacecraft and collapsing the navigation requirements for Doppler and Range into shorter tracks, we believe spacecraft can be adequately serviced using short 10 to 30 minute tracking sessions. This claim assumes that certain changes would have to be made in the way the Network traditionally services missions in Cruise. Furthermore, limiting spacecraft to short sessions will free up larger blocks of time in the tracking schedule to help accommodate future tracking demands soon to be placed upon the Network.

This paper describes the key characteristics and benefits of MMO, the operational scenarios for its use, the required changes to the ground system in order to make this approach feasible and the results of two simulations: 1) to determine the effects of MMO on projected mission loading on the DSN and, 2) to determine the effect MMO has on spacecraft orbit determination.

***Keywords**: Message Mode Operations, Adaptive Spacecraft Scheduling, Orbit Determination*

Introduction

The purpose of Message Mode Operations (MMO) is to utilize DSN tracking time and ground resources more efficiently by making the spacecraft an active partner in the scheduling process. MMO utilizes short 10 minute (minimum) to 30 minute (maximum) unscheduled periods of Deep Space Network (DSN) tracking time based upon spacecraft need and ground availability to determine the health and welfare of the spacecraft, download key engineering telemetry status, determine future scheduling opportunities, uplink a new ground schedule, and obtain necessary Doppler and range data. The purpose of this paper is to introduce and define the concept of MMO and to generate significant preliminary evidence to demonstrate MMO's usefulness as an operational mode for future spacecraft which utilize the services of NASA's Deep Space Network.

Philosophy of Use during Cruise

MMO is applicable during those portions of a mission's cruise phase in which no major activities are scheduled. For the candidate missions under study, this equates to several months during which spacecraft could operate in Message Mode, see Table 2. Activities such as instrument calibrations, and maneuvers which require longer tracking passes and two-way tracking are outside the scope of MMO.

Philosophy of Use during Extended Mission

MMO may also be used during the extended mission, since like the majority of the cruise phase, no major activities are scheduled. MMO telemetry could provide a science preview capability by means of thumbnail sketches of science opportunities for evaluation by the ground or on-board. Promising opportunities could result in the spacecraft generating a service request for a future tracking support.

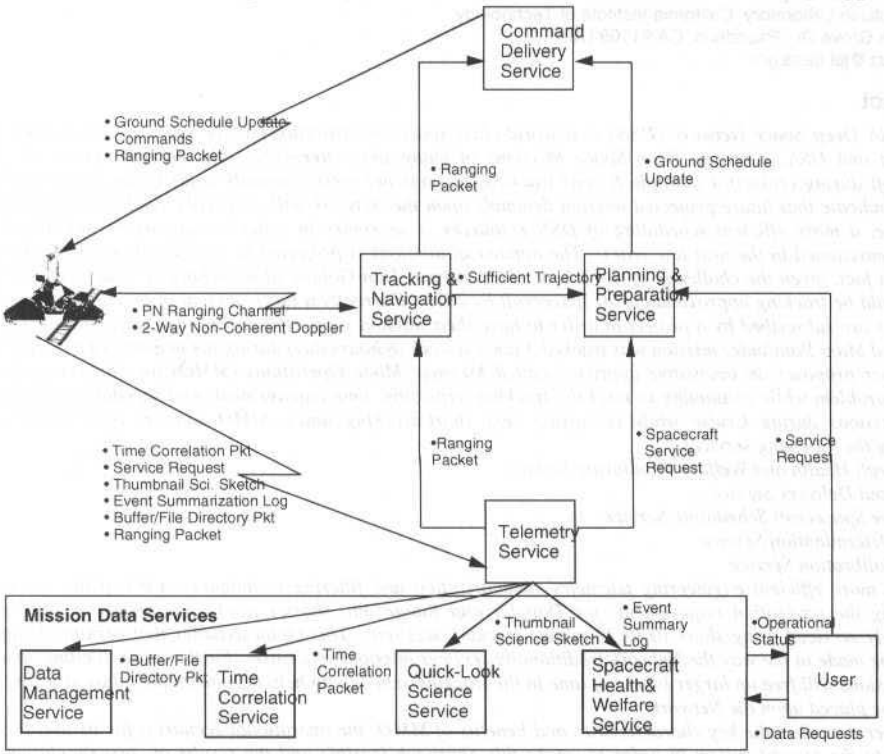


Fig. 1 Context of Message Mode Operations

Message Mode Operations Scenario

Figure 1 depicts the key functionality, data flows, and ground services involving Message Mode Operations. There are two major techniques for MMO spacecraft to communicate with the ground: hunting mode, and scheduled mode. The technique chosen will be a function of how frequently a spacecraft can transmit its MMO telemetry due to spacecraft resource constraints and how available the ground system is for tracking MMO spacecraft. The first technique, called hunting mode, is unscheduled and attempts to make use of short uncommitted antenna time that would normally go unutilized. In this mode, the ground attempts to make contact with multiple MMO spacecraft in a limited region of the sky over a 20 to 40 minute period. These MMO spacecraft have been receiving updates to the ground schedule and, therefore, have a model of when the ground is most likely to listen in including potentially the ground search priority. The second technique involves scheduling MMO tracks for those spacecraft whose mission needs require a guaranteed contact time. It is believed that most spacecraft could make use of hunting mode, given the reduced activity profile of most missions in Cruise.

The spacecraft and ground maintain a two-way non-coherent link from which both one-way Doppler and *coupled non-coherent range* are acquired, since an MMO track is too short to acquire a spacecraft in two-way coherent mode. The Tracking and Navigation Service processes these radiometric data and produces a trajectory sufficient to meet the less demanding tracking requirements of the spacecraft being serviced during Cruise or during extended mission. See the section, *MMO effect on Orbit Determination* for more details.

The trajectory is used by the Planning and Preparation Service as the trigger to produce the view periods file, frequency predicts, light time files, etc. required to support the tracking of the spacecraft. This service also produces the ground tracking schedule for all spacecraft, as well as receives requests for service from MMO spacecraft and from users. Service requests from the spacecraft and ground users may affect the current ground schedule and, therefore, a provision is made for uplinking future schedules and changes to these schedules via the Command Delivery Service. MMO spacecraft must be capable of accepting these updates during any uplink. The spacecraft validates these changes to the ground schedule and stores them on board as ground session windows. In general, commands, if necessary, as well as the ranging packet, providing the time tag when the last range measurement was received by the ground, along with the phase of the pseudo-range code at that time instant, are also uplinked.

On the downlink, the spacecraft telemeters all essential information in order for mission operations to 1) ensure the correct trajectory of the spacecraft, 2) maintain the time correlation between spacecraft time and UTC, 3) field service requests from the spacecraft indicating extended or future supports, 4) examine thumbnail science sketches providing the opportunity for quick look data interpretation and further science harvesting by issuance of a service request for further tracking, 5) monitor spacecraft buffer and/or file directory status via the telemetered buffer/file directory packet (Operations uses this directory to make decisions on what files need be downloaded during future tracks), 6) interpret the health, welfare, and state of the spacecraft by viewing the event summarization log, 7) provide to both the ground system, as well as the spacecraft, the capability of calculating the one-way range measurement from the range packet.

Table 1 Roadmap to MMO

Key Functions	Current DSN Design/Performance	MMO Enabler	Required MMO Design/Performance
Pre and Post Calibration	60 minute pre-calibration 15 minute post-calibration Range pre-calibration predominates.	Use the same exciter, transmitter, receiver with an antenna. Also, limit the range calibration to 3 range points.	Ranging Calibration is not required for each pass as long as same uplink and downlink elements used.
Spacecraft - Ground Scheduling	Spacecraft is a passive entity in the scheduling process. Ground schedules the spacecraft.	Spacecraft becomes an active partner in the scheduling process.	Automated Ground scheduling system uplinks scheduled tracking sessions and future potential sessions. Spacecraft provides times it has scheduled for other activities, reviews uplinked proposed schedules and downlinks service requests.
Orbit determination during Cruise	Nominally provides two-way Doppler to an accuracy of 0.1 mm/sec @X-band. Nominally provides two-way range from 1.0 m to 50 m accuracy @X-band.	Coupled non-coherent ranging. Relaxed position and velocity requirements during cruise. Reliability of Range points are 0.999.	1 or 2 accurate one-way range points collected per pass are sufficient to predict trajectory to 1000 km.
MMO capable spacecraft	Closest thing to MMO is Beacon mode currently being prototyped on DS-1, in which 4 distinct subcarrier frequencies signals 4 separate spacecraft states.	Demonstrated MMO ground system capabilities.	Spacecraft must accept and store ground schedule, updates to that schedule, and generate its own service requests.

Roadmap to MMO

The key functions within the current DSN that require change in order to enable MMO are: 1) Significant reduction in pre and post calibrations, 2) spacecraft-ground scheduling, 3) one-way

radiometric data types and their weighting for orbit determination during Cruise, and 4) the creation of autonomous MMO capable spacecraft.

The current system requires a long pre and post calibration period (see row 1, Table 1) in order to prepare a station for a track and ensure that the equipment remains within specification after the track. The short MMO track practically limits pre and post calibrations to 30 minutes total. As the station moves towards more automation through the service system design process, it is believed that both pre and post calibrations will be reduced to less than 30 minutes total. It is critical to note that the pre-calibration period is largely driven by the calibration of the ranging system. Alternative methods for doing range calibrations, such as calibrating during the track and reducing the time of a range measurement to 1 minute should enable the Tracking and Navigation Service System to achieve the short goal.

The current system provides a forum working within the resource allocation process (RAP) in which missions negotiate for tracking time amongst themselves. Long term schedule conflicts are resolved between mission representatives during these meetings. The system produces short term (7 day) and long term (8 week) schedules that for the most part mirror the agreements made during the RAP process. MMO requires a much more automated and flexible scheduling system. See row 2, Table 1 below. In order for the short duration MMO passes to be useful, the system is required to find unscheduled periods in an automated fashion. Moreover, the spacecraft becomes an active partner in the scheduling process informing the ground when it needs a track by means of a service request.

Orbit determination is largely accomplished for most missions today based upon two-way coherent communications (Doppler and range) in which the phase of the downlink signal is uniquely determined by the received uplink signal from the ground system. Generally, even in Cruise, the accuracy to which these measurements are made are typically well within the required tolerances of the missions. See row 3, Table 1. The short duration of the MMO, track coupled with the time required to pull the spacecraft oscillator to the required ground tracking frequency, necessitates that Doppler and range be acquired in one-way non-coherent mode. Fortuitously, a new technique, called *coupled non-coherent ranging*, has the potential for providing accurate range measurements within the drift specification of the on-board oscillator. See the section, *MMO Effect on Orbit Determination.*

Currently, spacecraft communicate with the ground under direction of the following 4 scenarios: 1) by immediate ground command, 2) by stored sequence time driven commands, 3) by fault response (emergency modes), or 4) the new beacon mode (see row 4 Table 1). None of these scenarios provides the capability for a spacecraft to automatically schedule itself into the ground tracking system. For MMO to provide for flexible on-demand scheduling, spacecraft will require the capability to submit service requests and receive ground schedule updates. This approach is in line with the Consolidated Space Operations Contract architecture in which spacecraft are to take full advantage of access on-demand and that utilize high levels of autonomy with minimum ground operations support in order to achieve expected cost savings.

MMO Effect on Deep Space Network Loading

The following is a preliminary assessment of how MMO affects the DSN tracking schedule. First, the mission subset over which the message mode operations scenario was applied is presented, followed by the mission cruise tracking requirements and assumptions applied in the study. Finally, the effect on the applicable DSN subnets i.e., 34M High Efficiency (HEF) and 34M Beam Wave Guide (BWG) as a result of applying Mission Mode Operations (MMO) to the scheduling system is reported.

A number of missions were provided as candidates for MMO. Of these missions, a number were not applicable due to limited tracking requirements in cruise. Following is a list of all the missions considered along with the missions included in the study clearly identified:

Table 2 Missions considered/included in MMO Loading Study

Missions	Launch	Mission Requirements	In study
Contour	7/2002	Hibernation mode during cruise	No
Deep Space 2	1/1999	Consigned with Mars Polar Lander	No
Deep Space 3	12/2002	Requirements insufficiently mature to Forecast	No
Genesis	1/2001	Cruise Period 5/01 to 3/04	Yes
Mars 01 Lander	4/2001	Cruise Period 5/10 to 11/01	Yes
Mars 01 Orbiter	3/2001	Cruise Period 4/01 to 8/01	Yes
MUSES-C	1/2002	Cruise not specified	No
Stardust	2/1999	Cruise throughout study period	Yes

Cruise requirements of the above listed missions were carefully chosen to include those segments that are truly in cruise mode. This means that trajectory correction maneuvers and other activities of a priority higher than Cruise were not considered within a MMO scenario. However, simplifying assumptions about MMO cruise activities would have to be revisited when the detail requirements of the cruise phase of these missions become available. The following assumptions were made concerning the MMO cruise phase of operations:

1. Independent of the current cruise phase requirements, it was assumed that during a MMO cruise phase, at least one contact per day would be required.
2. Improvements in pre and post calibration procedures would conceivably reduce the total calibration time to 30 minutes.
3. The MMO daily contact was assumed to be 30 minutes long.
4. There was no provision made for periodic long tracks during lengthy cruise operations. (This would require detail requirements negotiations with missions operating in the MMO mode.)

Given the assumptions defined above, the JPL Telecommunication and Mission Operations Directorate (TMOD) Resource Allocation Teamís FASTER suite of forecasting tools was used to evaluate the effect on the DSN resources given the assumptions for the one MMO scenario above. The following is a description of the results:

Forecast year 2001 was impacted the most by MMO technology. All selected projects operated in cruise phase for some portion of that year. Each mission's cruise scheduling requirements are unique, which explains the difference in the figure of merit, *increase in antenna time* in Table 3 below. This figure refers to the average increase in antenna time during that mission's cruise period, as a result of it switching from standard tracking to MMO technology e.g., if the Mars' 01 Lander switched to MMO, there would be a 55% increase in 34M BWG time over standard operations, and a 45% increase in available 34M HEF time.

Table 3 MMO Loading Effect on each Mission

MMO Mission	Increase in Antenna Time	DSN Subnet
Genesis	16%	34M BWG
Mars'01 Lander	55%	34M BWG
Mars'01 Lander	45%	34M HEF
Mars'01 Orbiter	35%	34M BWG
Mars'01 Orbiter	21%	34M HEF
Stardust	22%	34M HEF

For the applicable MMO study missions below in Table 4, the figure of merit represents the average increase in antenna availability for all missions for the given subnet over these mission's collective cruise periods. This metric provides a means for monitor-ing the overall effect MMO has on each subnet as more missions transition to this technology.

Table 4 MMO Loading Effect on each Subnet

DSN Subnet	Increase in Antenna Time	MMO Missions Included
34M HEF	9%	Mars '01 Lander
		Mars '01 Orbiter
		Stardust
34M BWG	16%	Genesis
		Mars '01 Lander
		Mars '01 Orbiter

Table 5 is the result of running the results of the loading study against the NASA JPL DSN station cost calculation form. The purpose of this exercise was to determine the total tracking hours and associated cost required to track the candidate missions in the study for a given aperture size. The results show an approximate 2.6x reduction in tracking hours for the Genesis mission and a cost savings of $88K, if MMO is used during cruise. For the Mars Surveyor 2001 Lander, tracking time was reduced by approximately 4x, with a cost savings of over $245K. A similar result occurred for the Mars Surveyor 2001 Orbiter. Stardust, on the other hand, showed a total reduction of 8 tracking hours, but an increase in tracking costs of approximately $100K. This is the result of the 7x increase in the number of pre and post-calibrations needed to support MMO 7 days a week as opposed to one six hour track of standard operations during cruise. It is hoped that through experience with MMO a reduction in MMO tracks to every other day or potentially one or two contacts per week can be achieved. Overall, the total delta cost

savings due to reduced tracking time during Cruise is approximately $0.5 Million dollars for these 4 missions alone.

Table 5 Comparison of Antenna Time and Cost in 2001 (MMO vs Standard Operating Mode)

Mission & OPS Mode	Total	Total Cost	Delta Cost
Phase	Time Reqd.	for period	for period
(name)	(hours)	(real-year $)	(real-year $)
Genesis - MMO	91	$107,748	
Genesis - STD	240.5	$195,775	$88,026
Mars 01 Lander - MMO	105	$124,325	
Mars 01 Lander - STD	416.25	$369,645	$245,320
Mars 01 Orbiter - MMO	112	$132,613	
Mars O1 Orbiter - STD	444	$394,288	$261,674
Stardust - MMO	231	$273,515	
Stardust - STD	239.25	$177,052	($96,463)
Total Delta Cost			$498,558

MMO Effect on Orbit Determination

Traditionally, orbit determination for deep-space missions has been performed using 4 to 8 hours of tracking per pass with two-way coherent Doppler. In many cases, two-way coherent range measurements are available if the spacecraft transponder design has the capability to receive and transmit a ranging code modulated on the carrier. Optical images of the target body taken by the spacecraft are sometimes used during the final weeks or months leading up to encounter. These optical navigation images are especially powerful if the target's ephemeris is not well known, as is the case with asteroids and comets.

There are generally two requirements imposed on orbit determination for each mission. The first and most important requirement is ensuring that the spacecraft is delivered to the target body within an acceptable level of error sufficient to meet mission objectives such as orbit insertion, flyby reconnaissance, or atmosphere entry and landing. This requirement varies from mission to mission. The second requirement is that at any time during the mission, the spacecraft's ephemeris is known well enough to allow ground stations to correctly point and acquire the spacecraft's signal without searching the sky or sweeping through the spectrum for the downlink carrier frequency. For missions trans-mitting at X-band, the requirement for accurately pointing the ground antennas is 130 arc-seconds (3-sigma), translating to a maximum position error of approximately 63,000 km at 100 million km distance from Earth[1].

Message Mode Operations would replace the 4 to 8 hours of two-way Doppler data collected per pass with only 10-30 minutes of one-way Doppler. Because in one-way Doppler the reference oscillator is onboard the spacecraft rather than on the ground, the frequency stability of the carrier is worse, resulting in Doppler measurements with 5 to 20 times more noise. Furthermore, onboard oscillators tend to drift with time due to temperature fluctuations and age, so this error must be estimated and removed to avoid obtaining an incorrect orbit solution.

Ranging can be performed with a non-coherent link between the ground and the spacecraft. The ground transmits a range code to the spacecraft, the spacecraft demodulates the code, records the time the range code was received, and re-modulates the signal back on the downlink while transmitting the code receipt time via telemetry. Once the ground receives the re-modulated range code, it has a measurement of both the uplink and downlink delay, and thereby a measure of range (for both upleg and downleg) is obtained. The accuracy of this ranging method measurement is dependent on the onboard clock error, therefore it is important that the spacecraft and ground clocks be synchronized, and the timing difference between them be estimated in the orbit determination process. This ranging technique, dubbed *coupled non-coherent ranging*, has yet to be demonstrated on deep-space missions.

To gauge the effectiveness of orbit determination in Message Mode Operations, a scenario was developed using the Mars Surveyor 2001 Lander as a test case. The Mars '01 Lander will be launched in April of 2001, and arrives at Mars in January 2002 after 9 months of flight and 5 midcourse corrections[2].

Because this mission will employ precision landing techniques for the first time at Mars, it is vital that the flight path angle upon atmosphere entry be kept to a minimum. The baseline navigation plan calls for two-way coherent tracking throughout cruise, starting with 3 passes per day for the first 8 days, then decreasing to 3 passes per week for 7 months, then returning to three per day 60 days before arrival. All passes are a minimum of 4 hours each, collecting two-way coherent Doppler and range every 10 minutes from the 34-meter stations located in Australia, California and Spain.

For the MMO case, all passes from Launch+8 days to Landing-60 days were reduced to only 30 minutes each per day. During these passes, one-way Doppler and coupled non-coherent range was simulated with noise values of 1.0 mm/sec and 100 m. respectively. The tracking data at the beginning and end of cruise (first 8 days and final 60 days) remained unchanged from the baseline case.

Figure 2 is a logarithmic plot showing the results. The two lines at the bottom are the root-sum-squared (RSS) position uncertainty resulting from orbit determination during cruise for both the baseline and Message Mode Operations cases. For each data point on the graph, tracking data up to that time is processed, and the resulting orbit determination error is mapped to the Earth-centered J2000 coordinate frame at that time. The gray line at the top indicates the maximum tolerable error in position in order to correctly track the spacecraft. As can be seen, the two cases start with identical uncertainties during the first 8 days of cruise where two-way data is available in both scenarios. At the second point in the graph, the MMO scenario starts using one-way data and, as expected, the uncertainty grows with respect to the baseline case. The maximum uncertainty occurs near September 18th, when the MMO case has an uncertainty of 850 km, yet this uncertainty is well below the maximum error requirement. Therefore, this figure indicates that Message Mode Operation tracking provides sufficient data for orbit determination for routine tracking.

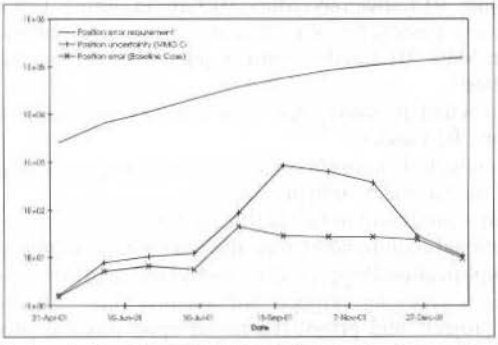


Fig. 2 Orbit Determination Uncertainties During Cruise for the Mars 2001 Lander

Figure 3 shows a B-plane projection of the orbit determination error 30 days before encounter for three cases; the baseline case, the MMO case (entitled "MMO case: two-way near encounter"), and a version of MMO that includes no two-way tracking ("MMO case: one-way near encounter"). This figure is meant to illustrate how confidently each mode can deliver the spacecraft to the desired target for atmosphere entry. The baseline case provides the smallest uncertainty ellipse (45 km x 10 km) since it uses the highest quality of tracking data and at greater quantity. The MMO case with two-way near encounter is approximately 50% longer (75 km x 10 km), and the MMO case with one-way near encounter is much larger (350 km x 10 km).

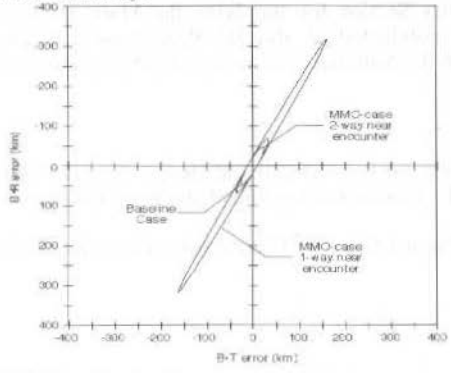


Fig. 3 Orbit Determination Results at Mars-30 days mapped to B-plane

The consequences of the largest of the three ellipses to the Mars '01 Lander mission would be a significantly higher probability of entering the atmosphere too steep and impacting the surface too hard, or entering too shallow and missing the landing site, or worse, skipping out of the atmosphere completely. Therefore, while brief, 30-minute one-way tracking passes available through MMO are sufficient for routine navigation during cruise, MMO alone is insufficient to satisfy the tight delivery requirements needed for missions like the Mars '01 Lander. In order to meet these delivery requirements, MMO would have to be followed with a traditional campaign of two-way tracking, or augmented with optical navigation images near encounter.

It's important to mention that one of the reasons current missions require long passes of tracking data is that, for various reasons, the navigation analysts typically discard 10-20% of the data as unusable for orbit determination. Since MMO yields far less data, each data point is therefore more "valuable" in the MMO case. The tracking system of the future will therefore need to be improved to provide a much higher tracking system reliability, so that less than 1% of all tracking data are considered unusable.

Future studies of MMO on orbit determination could examine the effect of reducing the number of passes per week, reducing the length of a MMO pass from 30 minutes, and also adding an Ultra-Stable Oscillator (USO) onboard the spacecraft to improve the Doppler quality.

Conclusions

The following conclusions are drawn from this study on MMO:

- It appears that tracking savings up to $0.5 Million dollars can be achieved if MMO is utilized for Genesis, Stardust, Mars '01 Lander and Mars '01 Orbiter during their cruise phases.
- It appears that extensive passes, i.e., 4 to 8 hours of two-way coherent radiometric data, are not required to meet the Mars '01 Lander cruise tracking require-ments, provided a technique like MMO can be substituted.
- MMO alone is insufficient to satisfy the tight delivery requirements needed at encounter for missions like the Mars '01 Lander.
- There appears to be enough of a cost benefit to merit a complete study of MMO by JPL's TMOD for future spacecraft such as the X2000 program.
- If cost and time savings are found to be significant for the majority of future missions along with meeting their navigational requirements then the next series of functional requirements on JPL's TMOD Network Simplification Project (NSP) should reflect MMO requirements.
- JPL's DSN will begin charging projects for antenna time. In the past, antenna time was negotiated between projects and prioritized based upon mission phases and emergency needs. Prioritization of antenna time is now a factor of economics as well as mission needs. In general, MMO may allow for tracking of more missions, since it appears to make better use of antenna tracking time. Therefore, it may aid in sustaining the expected increase in the mission set.

Acknowledgment

The authors would like to acknowledge the work of Dwight Holmes of JPL's Resource Allocation Planning and Scheduling Office for running the loading study, Gary Spradlin of JPL's TMOD System Engineering group for his assistance in developing the MMO concept, and Robert Mase of JPL's Navigation and Flight Mechanics Section for supplying the Mars '01 Lander information. The work described in this report was conducted at the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

References

[1]Deep Space Network System Functional Requirements and Design: Tracking System (1988 through 1993), DSN Document 821-19, Rev. C (JPL D-1662), Jet Propulsion Laboratory, Pasadena, California, 15 April 1993, pp. 3-27.

[2]Mars Surveyor 2001 Navigation Plan and Trajectory Characteristics Document (Preliminary Version), JPL Document D-16001, September, 1998.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 72

ISSN 0100-7386
Printed in Brazil

System Automation

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 73-83

ISSN 0100-7386
Printed in Brazil

High Level Tools for Satellite Operations

Roger S. Thompson
John A.R. Bone
Science Systems (Space) Ltd.
5 Greenways Business Park, Chippenham, Wiltshire, SN15 1BN, UK
roger.thompson@scisys.co.uk

Abstract

The traditional approach to satellite control relies on human controllers monitoring status displays and performing repetitive pre-set tasks. However, increasing demands are now being placed on satellite operations due to the increasing number of satellites under control, the increasing complexity of individual satellites, or the financial pressures of low-cost missions. In response to this, the agencies and corporations that operate satellites are looking to more advanced techniques to assist in the automation of day-to-day activities in the modern satellite control centre. This paper describes ***Space UNiT*** *(Universal Intelligent Toolkit), a suite of products that can be incorporated into both new and legacy mission control systems.* ***UNiT*** *supports the automation of operations procedures, contingency handling and schedules (timelines). It also provides the tools needed for off-line mission planning and procedure authoring.* ***UNiT*** *displays are highly graphical and support the monitoring and control of operations at a higher level.*
Keywords*: Mission Control, Satellite Control, Automated Procedures, Mission Planning and Scheduling.*

Introduction

Increasing demands are being placed on satellite operations due to the increasingly competitive nature of the industry and ever-expanding population of satellites. Advances in communications and software technology have lead to more sophisticated ground segment equipment that must also be controlled and monitored effectively. In an attempt to control escalating costs those responsible for operating satellite systems are demanding advanced tools to support a higher degree of automation. This trend is observed across missions with a wide range of characteristics:

- complex earth observation or science missions.
- "constellations" or multiple satellite systems.
- "smallsats" or other low-cost missions, for which the cost of operations is significant.

The **UNiT** product has been commercially developed by Science Systems from a demonstration prototype, which was originally produced under a joint ESA-Industry technology development programme: the advanced component of ESA's SLICK (Satellite Low Cost Integrated Control Kernel) project was the forerunner for **UNiT**.

The use of COTS (commercial off-the-shelf) products is seen as one way of achieving lower cost operations. COTS products generally entail lower initial costs than bespoke systems, this associated with the reduced maintenance overhead and support of COTS software makes the use of COTS products extremely attractive.

Based on a generic model of operations, **UNiT** provides a comprehensive system for the automation of mission operations. **UNiT** comprises a suite of co-operative components, which can be combined to provide varying levels of automation for both satellite mission control systems and checkout facilities.

The goal is to offer a generic solution for building highly automated satellite mission control systems, which provide an easy to use interface for both operations engineers during operations preparation and controllers during operations execution. The approach is to abstract operations to a higher level within the automated system, allowing an Operator to handle the control of more satellites and ground systems and thus reduce mission costs.

It is recognised that many existing mission control systems could benefit from increased automation, and consequently **UNiT** has been designed as a layered product, which operates through an underlying mission control system. This allows **UNiT** to be used in conjunction with a number of Satellite Control Centre (SCC) COTS, or overlaid on an existing mission control system.

UNiT Model of Operations

UNiT provides an environment for the specification of a mission operations model, reflecting both the current status of the system under control (spacecraft and ground segment), and the current status of mission operations. The latter includes automated operations support functions at three levels: *schedule*, *procedure* and *action*.

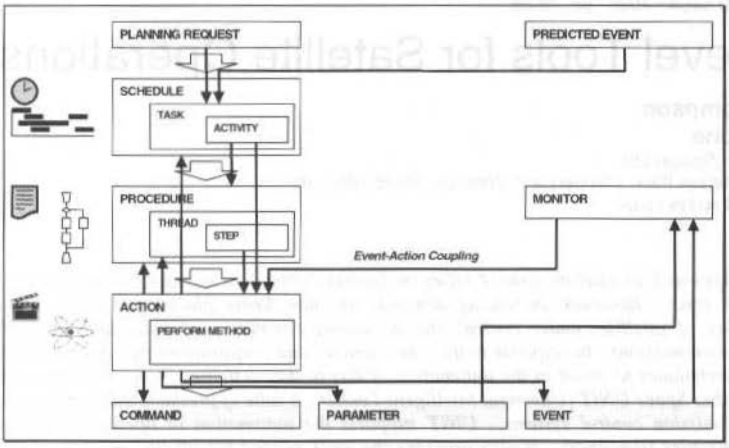


Fig. 1 UNiT Model of Operations

Fig. 1 illustrates the model used within the **UNiT** system. The model is split into horizontal layers and vertical columns. Each layer represents a level of abstraction: the Scheduled Timeline at the top, predefined Procedures or Monitor functions in the middle, and atomic Actions at the bottom. Each column represents different hierarchical models used to represent operations by **UNiT**, these being the Control, Status and Event hierarchies.

UNiT displays also follow this approach of layered abstraction. At the procedure level a status summary of all active procedures is shown, while individual procedures may be displayed in animated flowchart form. At the schedule level, the active schedule is displayed as an animated GANTT chart.

UNiT Interface Classes

At the foot of each hierarchy is a class of fundamental objects, which provides the interface to the underlying mission control system: *commands* (control), *parameters* (status) and *events*.

A command can equate to any control directive supported by the underlying mission control system (telecommand, service request, ground station command or control system directive). Each issue of a command constitutes a new instance of a transient command object, which may then be asynchronously updated to reflect the evolving status of that command instance until it is deemed to have completed.

Similarly a parameter equates to any status point maintained by the underlying mission control system, whether directly telemetered by the system under control, derived within the mission control system or relating to the control system itself. Parameters are persistent objects, which are updated asynchronously as each sample is received. Parameter value, availability and check status are all considered attributes of the same parameter object.

Events represent asynchronous alert conditions (e.g. messages and alarms) detected by the underlying control system, and not simply the change in status of a parameter. Like commands, events are instantiated with each occurrence of the event.

UNiT creates *command*, *parameter* and *event* interface objects by reference to the databases or definitions maintained by the underlying control system.

Actions

Actions are the atomic level of control within the **UNiT** operations model and constitute the first layer of control abstraction. They may be initiated directly from an *activity* scheduled on the operations timeline, from the body of a pre-defined operations *procedure*, or as the result of a monitored contingency via an *event-action coupling*. Various classes of *action* exist to encapsulate differing detailed methods of execution, but all may be initiated in the same way within **UNiT**. Standard *action* classes exist to act on the **UNiT** interface classes, and on other elements of the **UNiT** operations model:

- Send a *command*
- Set a *parameter*, *argument* or *local variable*
- Raise an *event*
- Start a *procedure*

- Control a *thread*
- Control the *schedule*

The set of available *actions* can easily be extended to handle mission specific functionality by producing a new specialisation of the *action* class, with an appropriate execute method.

Procedures

UNiT *procedures* correspond to pre-defined operational or test activities, which can either be scheduled as an *activity*, or manually initiated. *Procedures* may also call other *procedures*, permitting their decomposition into smaller, more maintainable units, which can be used in the context of several operations.

The definition of a procedure comprises its public interface (including *arguments*), local variables and a set of *thread* definitions. Each *procedure* contains a *primary thread* and, optionally, a number of *secondary threads*.

Threads

Each *thread* constitutes an independent flow of control through the procedure. The *primary thread* corresponds to the main flow of control and is usually activated as soon as the *procedure* is started and continues to execute until the *procedure* terminates. *Secondary threads* support monitoring or contingency functions, which are performed in parallel with the *primary thread* within the context of the active procedure.

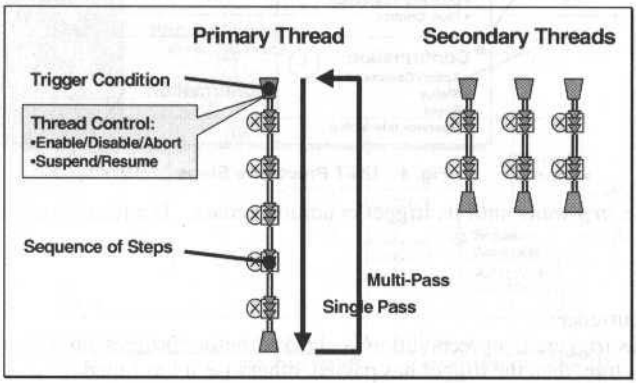


Fig. 2 UNiT Procedure Threads

Threads comprise a trigger condition and a sequence of *steps*. A *thread* activates following occurrence of its trigger condition and then proceeds to execute the *steps* in sequence.

Threads may also be defined as single or multi-pass: multi-pass *threads* reset themselves at the end and wait for their trigger condition to recur. This can be used to implement closed loop monitoring and control with hysteresis: e.g. thermal control – one thread to turn a heater on, another to turn it off.

Threads are represented graphically as flowcharts. They are defined using a graphical drag-and-drop editor, and animated during execution to display current status.

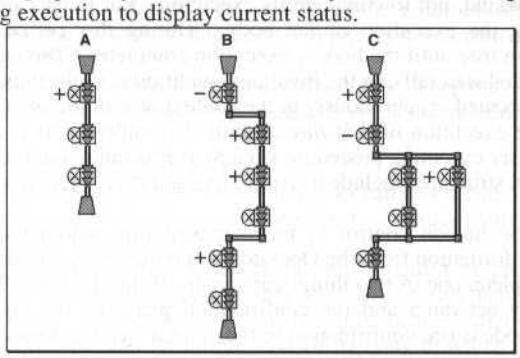


Fig. 3 Thread Outlining

The graphical representation of *threads* also supports outlining, as illustrated in Fig. 3. *Steps* can contain *sub-threads*, either to support flow control structures such as branches or loops, or simply to group a sequence of *steps* into a single outline *step*. These *sub-threads* may be expanded or collapsed using the +/- button that appears beside the *step*. This works in a similar way to the outlining features of MS Windows (explorer view). The figure shows three views of the same *thread*: fully collapsed outline (A); an outline *step* expanded to show the *sub-thread* (B); and a branch *step* expanded to show True and False *sub-threads* (C). Note that *sub-threads* can be nested.

Steps

All **UNiT** *steps* have a generic structure, with three phases of execution: trigger, body and confirmation. Each phase in turn has three elements (execution, checks and recovery) and four states (inactive [grey], executing [amber], passed [green], and failed [red]). The state of each phase is colour coded on the step icon and animated during execution.

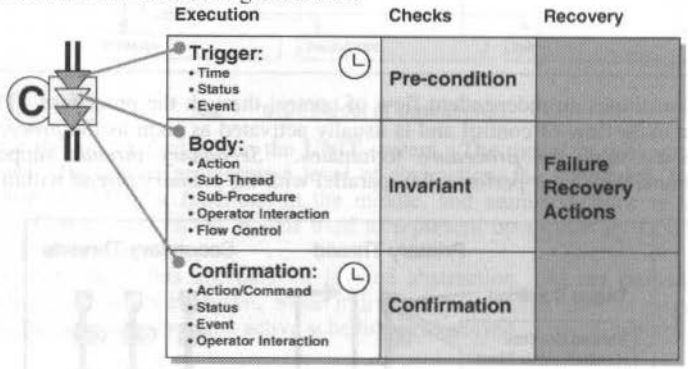


Fig. 4 UNiT Procedure Steps

On activation the *step* waits until its trigger condition occurs. The trigger may be expressed in terms of:

- Time
- Status
- an Event occurrence

Once the *step* has triggered, a precondition is checked before progressing on to the body of the *step*. If the precondition is true, then the trigger has passed, otherwise it has failed.

The *step* body itself may be:

- an *action*
- a *sub-thread*
- a *sub-procedure* call
- a user interaction
- a flow control structure

The execution of the body of simple atomic *action* is non-blocking and effectively instantaneous (the *action* is to initiate a command, not to complete its execution). For other categories of *step*, however, a finite time elapses during the execution of the body. During this period an invariant condition is monitored and must remain true until the body's execution completes. The execution of the body fails if the *action* fails, an embedded *step* fails, or the invariant condition is violated before execution completes.

A *sub-procedure* is executed synchronously in the context of a single *thread* of the calling *procedure*, and effectively blocks the execution of that *thread* until it completes. It is also possible to initiate an independent, asynchronously executing procedure via a Start Procedure *action*.

Supported flow control structures include IF-THEN-ELSE and CASE branches, various loops and parallel execution constructs.

User interactions allow human control to be integrated into automated operations, by enforcing check-points, requesting information from the Operator, or giving options to the Operator.

Once the body is complete, one of two things can occur. If interlocking mode is OFF, the next *step* in the *thread* is immediately activated and the confirmation phase of the current *step* is performed in parallel. If interlocking mode is ON, confirmation of the current step must succeed before moving on.

Confirmation itself requires a condition to be observed within a specified time window. The confirmation check can be expressed in terms of:

- the final state of an initiated Command
- Status
- an Event Occurrence
- Operator confirmation

Recovery actions can be defined to cater for failure of the *step* at any phase.

Monitors

Monitors are a lightweight free-standing, thread-like construct, which continuously look for an *event* or potentially complex status condition to occur, independently of any *procedure*. When this happens, a pre-defined *action* is executed. This mechanism is also referred to as an *event-action coupling*, and is used to monitor for known anomalies or other asynchronous conditions, and to automatically initiate contingency recovery procedures.

Note that *events* may also be used as interlocks within *procedures* when specified as *step* trigger or confirmation conditions.

Schedules

UNiT *schedules* constitute a timeline for automatic execution of planned operations or tests. They are not, however, fixed timelines, as they contain knowledge of the planning constraints and residual constraint windows, such that fine scheduling can occur during schedule execution to accommodate minor variations in the duration of *activities* and their associated resource usage.

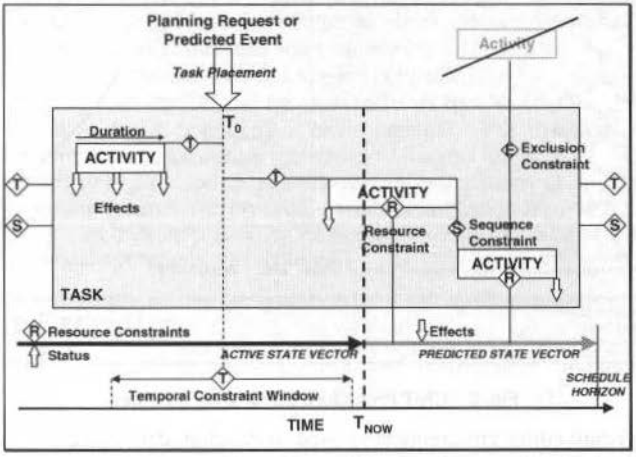


Fig. 5 UNiT Schedules

Schedules comprise a timeline of *tasks*, each of which corresponds to a discrete operation resulting from an individual planning request, or a predicted event (e.g. from Flight Dynamics). Each *task* has an associated pre-defined sub-schedule of *activities*, which can be automatically placed on the timeline. *Activities* correspond to individually schedulable items – typically a *procedure* or *command*, but in effect any *action* supported within **UNiT**.

Constraints can be defined at *schedule*, *task* and *activity* levels. They may be:

- temporal/sequential *constraints* (indicating when one operation should be performed in relation to others)
- resource *constraints* referencing a *state vector*
- overlap *constraints* indicating whether or not *activities* can be executed in parallel.

Effects are also associated with *activities* and correlate to resource *constraints*. They are used to propagate the *state vector* during scheduling.

A *task* has arguments, which originate from the planning request or predicted event, and may be referenced in any expression within the scope of that *task*: in the arguments of constituent *activities*, *constraints*, *effects* or *durations*. Expressions within the scope of an *activity* may in turn reference its arguments. In this way, arguments can be propagated all the way down to executable *procedures* and *commands*.

UNiT Applications

The **Space UNiT** product suite currently comprises the **UNiT** kernel, and two add-on modules: **SCHEDULE-iT** and **PLAN-iT**. Additional modules are planned for the future. The **UNiT** kernel supports the action and procedure layers of the **UNiT** operations model, and comprises an on-line procedure execution module and an off-line procedure editor and test-bed. **SCHEDULE-iT** and **PLAN-iT** support the schedule layer: the former providing automated timeline or schedule execution; and the latter off-line mission planning capability.

UNiT: Procedure Editor and Test-Bed

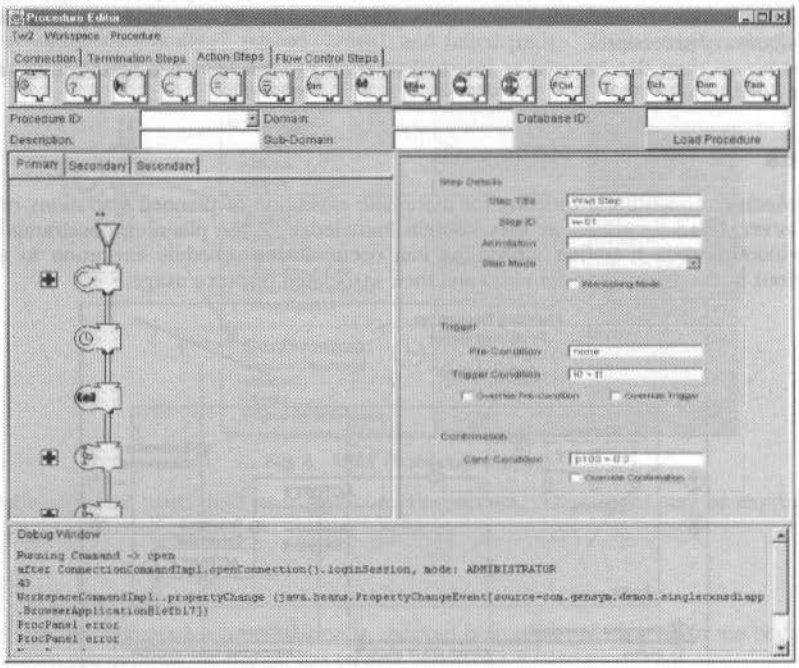


Fig. 6 UNiT Procedure Editor User Interface

The off-line procedure editor environment is used to develop, debug and test automated procedures. Procedures are developed using a graphical editor and a drag and drop approach. Databases containing the definition of commands, parameters and events supported by the underlying control system are linked into the editor. The outlining capability can be used to assist in the design and implementation of procedures in a top-down manner.

A totally portable Java client user interface is provided which gives a Motif or Windows standard look-and-feel dependent on the platform on which the client is running. The client offers standard menus, toolbars, widgets and dialogs generally associated with a modern user interface.

The graphical drag and drop approach facilitates the rapid development of procedures without a **UNiT** user having to understand any programming or scripting language. This same graphical representation of a procedure is used in the on-line environment to show the animated execution of a procedure.

The **UNiT** procedure editor also allows a user to define monitors and associated event-action couplings. Again these are represented in a graphical manner and stored in the procedure database.

The editor incorporates a test-bed facility consisting of the run-time procedure execution engine and an interface test harness. This enables basic procedure testing to be performed without the use of a full simulator.

UNiT can also provide a mechanism for the import of legacy text operations procedures into a graphical representation. These procedures can then be edited using a drag and drop approach and subsequently executed in the on-line environment.

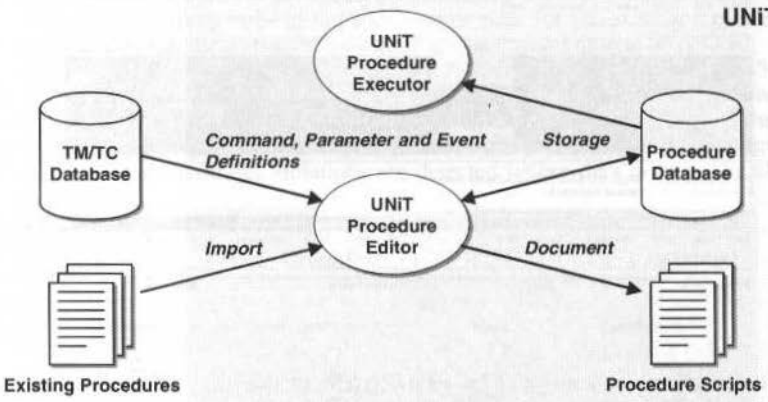


Fig. 7 UNiT Procedure Editor

Once the Procedure has been built and tested it is stored in a database external to **UNiT**. The public definition of the procedure is held as a relational database structure, and includes the definition of any arguments and lists of referenced objects (commands, parameters, events and other procedures). These lists are generated automatically by the **UNiT** procedure editor. This provides several key benefits:

- Consistency Checking – scripts can be written to check the consistency of procedures against other mission objects such as commands and parameters.
- Configuration Control – procedures can be version controlled.
- Open Storage – procedures can be read by other software besides **UNiT**.
- Separation between live and development environments – The database acts as the interface between the development and execution systems.

A Script Generation utility is provided to generate text representations of the Operations Procedures created by the **UNiT** procedure editor. The procedure's representation is read directly from the database the resultant text procedure can then be printed or viewed by a user. This allows a mechanism to be put in place if a fallback to manual operations was required.

UNiT: Procedure Execution

The **UNiT** Procedure Execution module is the run-time environment for procedures and supports their execution and display. A procedure may be executed in one of four ways:

- Manually by the Operator
- Automatically by another procedure
- Automatically by a monitor
- Automatically by a scheduled activity

When a procedure is to be executed, its definition is loaded from the procedure database. Included in the design is a caching mechanism, which stores a predefined number of procedures within the run-time environment in order to minimise data accesses for commonly used procedures. Once loaded, the procedure and its component threads, steps and actions are executed according to the selected execution mode. Two modes are supported: automatic and single step.

All active threads contained within the procedure are executed in parallel and in turn each step in the thread is executed in sequence. Whenever a state change occurs within the procedure the information regarding that state change is stored in the Procedure log file.

Debugging facilities available within the system allow single stepping through a procedure and the pausing of execution via the "current step" token shown alongside the active step in each thread. When the procedure is running in normal operation this appears green, when the procedure is paused it changes to red.

Once the procedure is complete the instance remains active for a configurable period and is then deleted. This allows the Operator to view the completed procedure, including the path taken through each thread and the final status of each step.

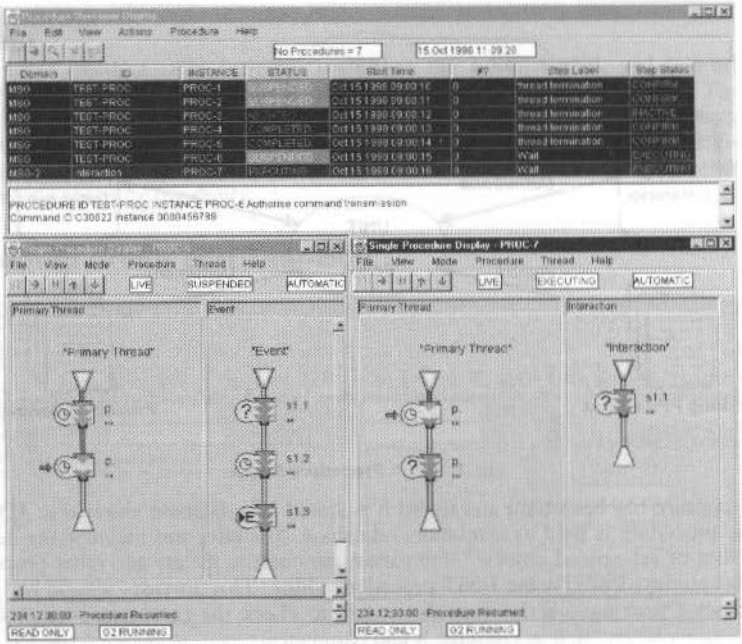


Fig. 8 On-line Procedure Display

SCHEDULE-iT: Schedule Execution

SCHEDULE-iT gives the Operator a high level view of the scheduled operations. These may be filtered and sorted using a number of criteria, including the "domain" of operations. A domain may correspond to a single satellite or ground segment facility.

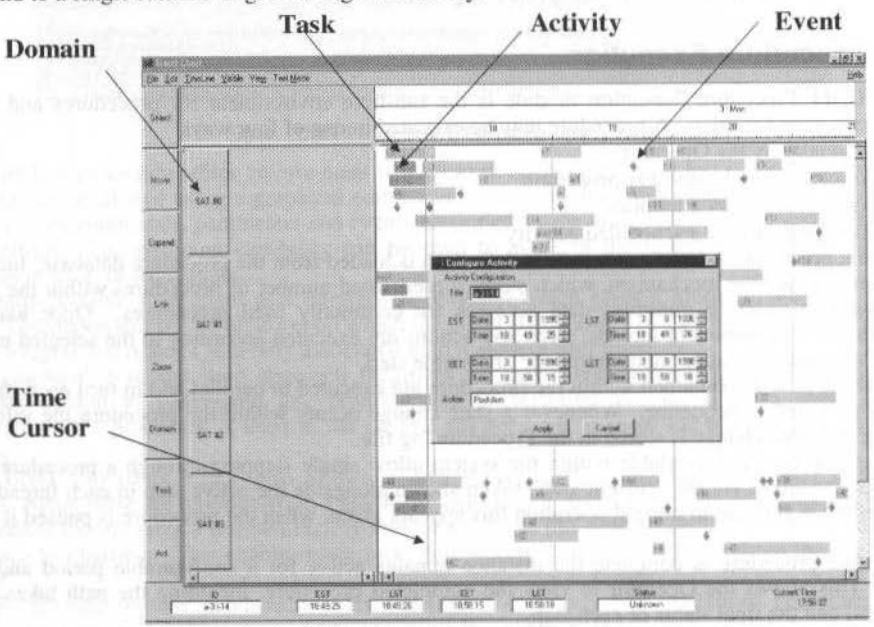


Fig. 9 Sample On-line Schedule Display

A schedule is created or edited via the Gantt user interface. New activities are defined manually by dragging and dropping an activity node at the approximate time for the execution of the procedure. A dialog is then displayed which allows the procedure ID and exact start time to be defined.

Once a schedule has been transferred from **PLAN-iT**, it can be loaded into SCHEDULE-iT. SCHEDULE-iT then executes the schedule, updating the Gantt display automatically to reflect the status of the scheduled procedures. During execution, SCHEDULE-iT performs constraint checking and fine scheduling against a predicted state vector. This is periodically propagated up to a configurable schedule horizon, such that potential constraint violations are detected before they occur. The Operator is then able to take avoiding action.

SCHEDULE-iT displays the following procedure statuses via the Gantt:

- Pending
- Executing
- Suspended
- Executed
- Failed

SCHEDULE-iT also allows the user to drill down by selecting an activity of interest and requesting the detailed graphical view of the associated procedure, as loaded by the Procedure Execution component of **UNiT**.

SCHEDULE-iT provides the means to define a schedule and view its execution via a Gantt chart interface. Execution of activities is handled via Procedure Execution but facilities are provided to navigate to a procedure instance via the Gantt. All control of the loaded (executing) schedule is via SCHEDULE-iT, it may be started and stopped at any time without impacting procedure execution.

PLAN-iT: Mission Planning

The **PLAN-iT** component of **UNiT** provides support to the full planning lifecycle and integrates with the **UNiT** kernel and schedule execution component (SCHEDULE-iT) to process operations through from initial requests to uplinked telecommands. PLAN-iT provides an environment within which users can specify, generate and refine plans and schedules for all satellites and ground segment facilities subject to control. The planning process begins at the highest level with sub-plans which group together the events and requests for a particular domain over a given time interval. The planner manipulates these sub-plans, instantiating tasks, activities and their respective constraints. An initial specification is expressed in terms of sub-plans and progressively refined, with the aid of a set of software tools, into an executable schedule.

PLAN-iT provides an environment for the development of executable schedules, from pre-scheduling through to refined scheduling. The user is able to lock elements of the evolving schedule as fixed points around which the rest of the schedule should be generated.

Conflict detection and resolution is achieved using a number of techniques within PLAN-iT. Conflicts may be resolved on a prioritisation scheme, through global optimisation strategies or through the application of local heuristic strategies. Conflict detection uses temporal, exclusion and resource constraints.

PLAN-iT complements other elements of **UNiT** including the schedule and procedure execution systems. In particular, a number of common components are shared with schedule execution; off-line mission planning includes state vector propagation and conflict detection.

PLAN-iT is a user driven mission-planning environment that supports the planner in the development of executable schedules. The user centred approach adds further weight to the importance of providing an intuitive user interface.

The user interface for planning supports the development of plans and schedules through from initial requirements, scheduling and the delivery of an executable schedule. The interface is built around a Gantt chart display, a range of editing facilities and a suite of scheduling tools.

PLAN-iT and SCHEDULE-iT share a modular architecture, which ensures open evolution. Well-defined interfaces mean that components are interchangeable, for example, the bridge to the scheduling engine could be replaced by a bridge to any other engine able to use the provided API . In this sense PLAN-IT is open in terms of its constituents.

The effects model used within PLAN-IT is extensible, users are able to extend the functionality of the effect expression language and to develop models of arbitrary complexity. PLAN-iT is open in terms of extensibility.

Status Distribution and Persistence

UNiT provides an API to allow the output of procedure and schedule status information to the wider mission control environment. This can be used for real-time distribution of status, and also for linking into an external historical archive. UNiT supports the logging of both procedure and schedule history. Where supported, retrieval of this historical data by UNiT can be used to provide historical procedure and schedule displays in both static retrieval and dynamic replay modes. In static retrieval mode, the final state of a procedure, or the executed schedule, is shown. In dynamic replay mode, displays update in pseudo real-time.

UNiT Integration

A major design driver for the **UNiT** product is to allow it to be integrated with any underlying mission control system. To this end a general purpose API has been constructed which allows the exchange of event, parameter and command objects.

UNiT was originally developed to run on a Windows NT platform, but was designed with portability in mind. It is also designed to be scaleable from a single computer to a distributed client-server network. **UNiT** itself is modular, to enable customisation and extension for specific missions. To enable this, an overall system concept and architecture has been defined which is sufficiently open to permit the integration of mission specific modules.

UNiT is based on a three-tier architecture that allows for the separation of the three key areas of Man Machine Interface, the application and the data the application uses. This architecture allows **UNiT** to be deployed on a variety of platforms, including Windows NT, Unix and Vax/VMS, in a distributed or standalone fashion.

The technology that enables this architecture is provided by Gensym's G2-Gateway product. As **UNiT** is constructed using G2 it has all the portability and integration advantages of G2. **UNiT** provides a generic API to enable the passing of event, command and parameter objects. A bridge process is constructed using a standard C++ library, which is linked to user specific code to obtain the required data from the external control system.

Fig. 10 shows an outline of the **UNiT** three-tier architecture. The figure shows that data acquisition and control is partitioned from the application, which again is partitioned from the MMI. From a **UNiT** system perspective this means that the MMI's could reside on a different machine to the applications which in turn could reside on a different machine to the Data servers. Conversely, all three layers could reside on the same machine, or any combination of machines.

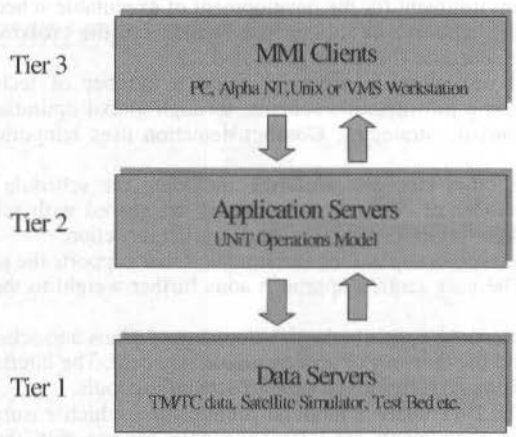


Fig. 10 UNiT Integration

The MMI's developed for **UNiT** have been constructed using Visual C++ and Gensym's Java Bean components. This enabled the production of a consistent windows based MMI. Science Systems is currently working actively with Gensym to extend this area of **UNiT**.

Database dependency is minimised through a data management API that uses standard database access mechanisms such as ODBC and JDBC. Platform independence for the user interface is supported through the use of the ILOG Views multi-platform windowing environment or optionally through Java.

UNiT Future Direction

Science Systems are proceeding with the development of additional modules in the **UNiT** product suite. Future modules are planned to provide support for the following functions:

- Pre-launch check-out
- Alarm filtration and fault diagnosis
- Network monitoring via SNMP
- Management of satellite constellations

Summary of UNiT Benefits

The use of the UNiT product suite offers the following benefits for both new and existing satellite mission control centres:

- Automation of Operations Procedures
- Automated Schedule/Timeline Execution
- Supports both Satellite and Ground Segment Automation
- Can support multiple "domains" – satellites or ground segment facilities.
- Graphical Procedure Development Environment
- High Level Control Displays: Schedule GANTT and Animated Procedure Flowcharts
- Open Interfaces: can be Integrated with any underlying Mission Control System.
- Can be overlaid on legacy systems.
- Can access existing databases defining telemetry parameters and telecommands.
- Existing procedure scripts can be imported.
- Can be deployed on Windows NT, Unix and VMS platforms.
- Can be used to support any level of automation from Operator Assistant to Lights Out Operations, providing scope for substantial savings in the cost of operations.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 84-89
ISSN 0100-7386
Printed in Brazil

Next Generation User Support Tools

Jeremy Jones
Tom Brooks
Lisa Dallas
Sandy Grosvenor
Anuradha Koratkar
LaMont Ruley
NASA Goddard Space Flight Center
Code 588
Greenbelt, MD 20771 USA
jeremy.e.jones@gsfc.nasa.gov

Abstract

One of the manually intensive efforts of Hubble Space Telescope (HST) observing is the specification and validation of the detailed proposals for scientists observing with the telescope. In order to meet the operational cost objectives for the Next Generation Space Telescope (NGST), this process needs to be dramatically less time consuming and less costly. We have prototyped a new proposal development system, the Scientist's Expert Assistant (SEA), using a combination of artificial intelligence and user interface techniques to reduce the time and effort involved for both scientists and the telescope operations staff. The Advanced Architectures and Automation Branch of NASA's Goddard Space Flight Center is working with the Space Telescope Science Institute (STScI) to explore SEA alternatives. We are testing the usefulness of rule-based expert systems to painlessly guide a scientist to his or her desired observation specification. We are also examining several potential user interface paradigms and exploring data visualization schemes to see which techniques are more intuitive. Our prototypes will be validated using HST's Advanced Camera for Surveys (ACS) instrument (scheduled for installation in 1999) as a live test instrument. Having an operational test-bed will ensure the most realistic feedback possible for the prototyping cycle. In addition, once the NGST instruments are better defined, the SEA will already be a proven platform that simply needs adapting to NGST-specific instruments.
Keywords: *NGST, SEA, rule-based systems, visual tools, proposal development.*

Background

The SEA is a prototype effort supporting pre-phase A development for NGST. The objective for SEA is to develop and test new approaches and tools to support the development of observing proposals. The NGST goal is to reduce the staffing requirements for NGST's proposal support staff by 50-80% as compared to HST. In order to accomplish this, the software support tools need to have a great deal more "intelligence" built into them to give observers the supporting guidance with little or no assistance from the telescope's staff. Our SEA team includes a small group of software engineers and astronomers from Goddard Space Flight Center (GSFC) and STScI.

Initial Analysis

During the first several months of the project, the team spent most of its time understanding the existing proposal process and tool set in use at STScI to support HST. We evaluated a series of potential tools and development environments. We developed a proposed strategy and tool set and a development and test schedule. Based on this analysis, the team believes that the combination of new visual tools and a rule-based expert assistant will have the biggest impact. Our driving philosophy is that the system must let scientists be scientists and not try to make them instrument or software engineers in order to develop their proposals. The tools should bring relevant reference material to the scientists' fingertips when needed. The tools should minimize duplication of effort, integrate information together, and point the way to defining good programs. The tools should spot problem areas early and be smart enough to recommend alternatives.

In addition to analyzing the existing processes, the team has developed a road map for developing and testing SEA. Since the SEA system will be almost completely new software, it is critical to have a realistic environment in which to perform final evaluation. Consequently, we intend to use ACS as an operational testbed. SEA is not currently intended to be the operational platform for ACS proposals, rather it is a parallel development effort. In doing so, we will have the full complexity of an operational environment. We can then compare the time and effort involved in developing a proposal using both the current HST tools and SEA.

Our main philosophy for development of the SEA is:

- The system should be *intelligent*. It should employ artificial intelligence methodologies and paradigms to assist and guide the scientist in producing a proposal that is flight ready.
- The system should be *intuitive*. The user interface should not require extensive training. Scientists should be able to use the SEA with little or no assistance.
- The system should be *distributed*. It should allow delivery and processing of proposals via the World Wide Web across a wide range of computing systems.
- The system should be *adaptable*. As the telescope staff learns how best to use NGST once it is launched and operational, the system should be able to incorporate new information and knowledge easily.
- The system should be *easily integrated* with other NGST planning and operations modules.
- The system should be *flexible*. Since NGST is not scheduled for almost a decade, the system development must allow for changes in technology. Further, much of the process of developing observing proposals is common among observing platforms. This system could and should be an effective alternative for other observatories, both present and future.

We have been looking at both the overall process and emerging technologies to see how the process can be better supported through automated means. We are currently in the middle of the development phase of this project. Once development is completed, we will begin the evaluation phase where the SEA will be evaluated against the current HST process and benefits will be quantified. In the current phase, we are developing several related modules and interfaces that will integrate the modules together. Throughout these modules we have several guiding principals driving the development:

- Integrate the various tools to minimize or eliminate duplicate entry of the same information. For example, once the instrument mode and targets have been specified, the scientist should not have to go to a separate exposure calculator tool, re-enter that information to obtain exposure times, and manually enter those exposure times back into the proposal. In addition, if the scientist has "jumped ahead" in a process and the system needs information that has not yet been specified, the system should ask for the information when needed (not just kick the user back to a different place in the software).
- Standardize the user interface for the computational tools across the different instruments in order to decrease the learning curve and increase efficiency.
- Thoroughly integrate reference material, providing an intelligent context-sensitive means to helping scientists find relevant technical reference material quickly and easily.

Progress and Plans

During the first phase of the project, the team focused on gaining an overview of the proposal process and defining the scope and schedule for the prototype effort. We reviewed the overall STScI process for detailing observing proposals (known as the Phase II proposal), and the areas within this process that are currently manually intensive. We decided on a design that divides the SEA into separate tools, or "modules" that are integrated into a cohesive framework, but can also be run separately if desired. This framework is represented to the user as the Proposal Browser window.

Proposal Browser

The Proposal Browser is the user interface framework for the SEA. This window resembles the Microsoft Windows Explorer interface, allowing the user to quickly navigate through the proposal by selecting elements in a tree view of the proposal. On the left, the contents of the proposal are represented as a tree. When the user selects an item in the proposal tree, the module for that item will appear on the panel to the right. The user may jump to any area of the proposal at any time. This technique allows only a single module to be open at once, similar to a page in a Web browser. To overcome this limitation, the Proposal Browser also allows the user to open a module in its own separate window by double-clicking on the item in the proposal tree, or selecting an "Open in New Window" option from the menu. This allows the user to have many different modules open at once. In this case, all modules are linked to the proposal such that if the user makes a change in one module, it is automatically propagated to all other modules.

The current prototype includes an implementation of the Proposal Browser. In addition, the following modules have been implemented and are accessible from the Proposal Browser:

Visual Target Tuner

The Visual Target Tuner (VTT) allows the user to visualize the area surrounding the target and to tune the target position. The VTT also allows the user to specify constraints for the orientation of the

observatory. These constraints are specified as objects or regions to include or exclude from the exposure. The SEA can then calculate the set of valid orientations given those constraints. The overriding design goal for the VTT is to create a highly visual and interactive environment for dealing with the position of the target. The user is able to easily move around the target area, zoom in or out of the visualization, identify objects on the visualization, and be able to drag the target to change its position. We are also experimenting with other features, such as field-of-view display, sky scanning patterns and survey layout, and a centroid feature. Our plans for the VTT include the ability to predict and visualize image artifacts such as diffraction spikes and CCD bleeding. A sample of the VTT is shown in Figure 1.

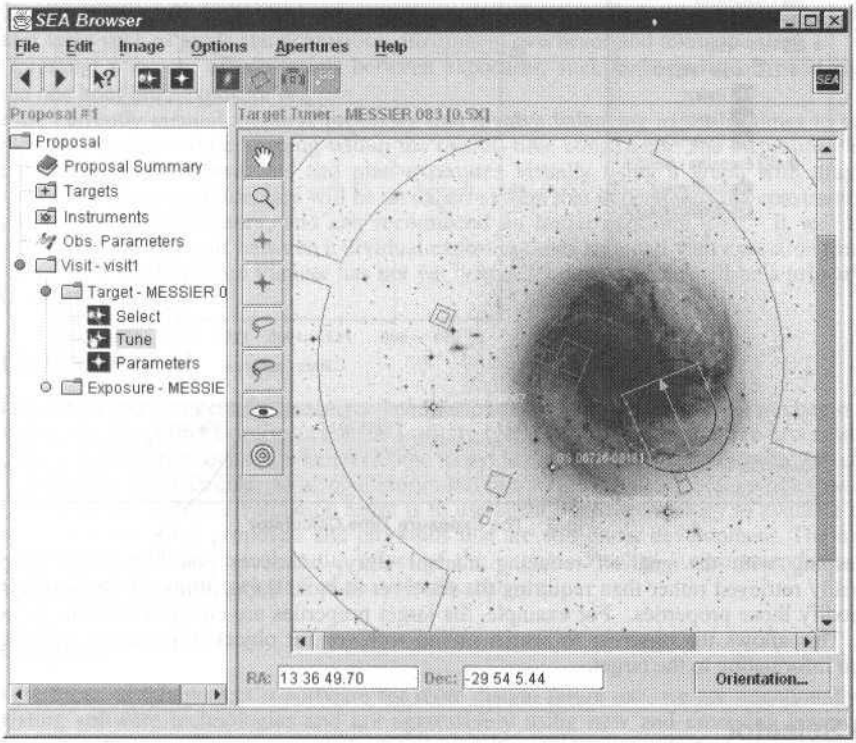


Fig. 1 The Visual Target Tuner

Exposure Time Calculator

The Exposure Time Calculator (ETC) generates real-time interactive graphs showing signal-noise ratios (SNR) and source counts across a range of exposure times and wavelengths. The user can manipulate the graph by zooming in or out of specific areas. The user can also ask the ETC questions like, "What exposure time do I need to obtain this SNR? ", or "How many counts will I get for this exposure time?" By providing this interactive capability, the observer is able to more easily experiment with exposure times and optimally configure their exposure.

The ETC also includes supplemental modules for Target Properties and Instrument Configuration. The Target Properties module allows the user to tweak the properties of the target model, including normalization, magnitude, point/extended source characteristics, and the spectral energy distribution model. Changes in the target model are instantly reflected in the ETC. Similarly, the Instrument Configuration module allows the user to configure the individual properties of the instrument, including the detector, filter, binning, etc. In addition, this module displays a graph of the throughput wavelengths of the current configuration. In later phases we expect to integrate the Exposure Time Calculator and the VTT so that a user can "see" a simulated image. This simulated image would predict what the observation would look like given known information about the target and the instrument such as data from previous images, characteristics of the observatory, detectors and filters. A sample of the ETC is shown in Figure 2.

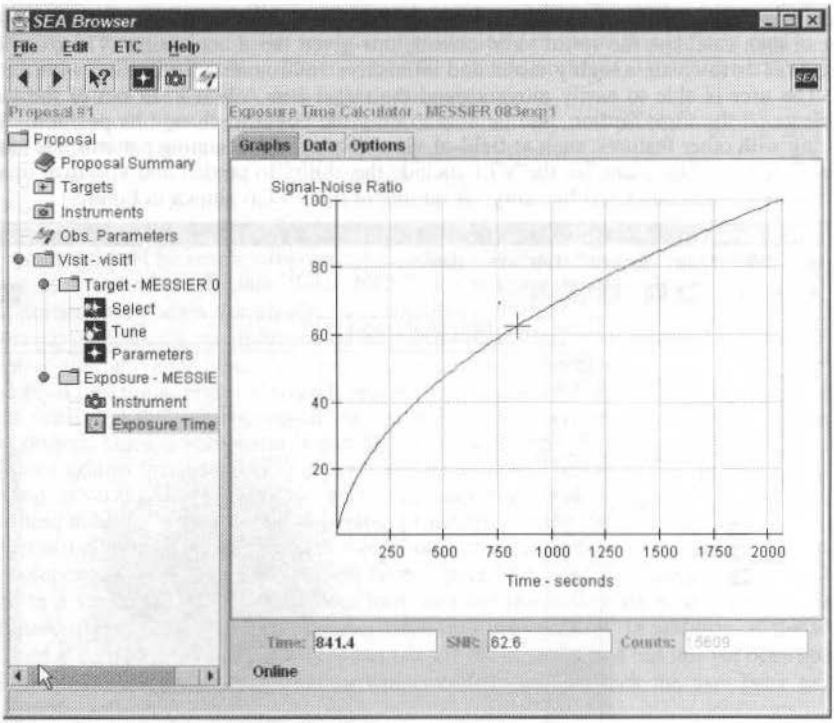


Fig. 2 The Exposure Time Calculator

In keeping with the goal of reducing manual entry, wherever possible, target properties are automatically retrieved rather than requiring the observer to enter them, although the observer is always free to modify those properties. For example, the target properties are connected to the Target Selector module. This allows the observer to search online archives for object information and automatically assign that information to the target.

Target Selector

The Target Selector is an interface to various online astronomical archives. It provides the ability to do simple object queries across a range of archives. Search results are displayed in a table, and the user has the option of assigning an object to a target. The current version provides limited query capabilities: searches by name, near name, and near position are supported. Currently supported archives include the NASA Extragalactic Database (NED), SIMBAD, and the HST Guide Star Catalog.

Instrument Configuration Expert System

The SEA allows the user the tweak the individual attributes of the instrument configuration. However, one of the overriding design goals of the SEA is to insulate the observer from the technical details of the instrument when desired. To accomplish this, the SEA uses an expert system to provide an interview question-and-answer session that will guide the observer through configuring the instrument. This feature is optional and is intended to accommodate those users who do not know or do not care to know the technical details of the instrument. This interview module attempts to ask the observer about the science requirements and then recommends instrument settings to accomplish those requirements.

We are currently in the process of building a rule base for the Advanced Camera for Surveys (ACS) instrument. The current prototype includes support for detector and filter selection. A later release of the SEA will include the ability to configure the entire ACS instrument using the expert system module. We are striving to discover if the tool can contain a sufficient level of science expertise to significantly reduce the support needed from STScI scientists.

In addition to the above modules, we are also planning the following new tools for 1999:

Exposure Planner

Thus far the modules described have focused on defining a single exposure. The Exposure Planner will work to provide assistance in laying out multiple exposures and planning groups of exposures (or "visits"). Both observers and STScI staff currently spend a great deal of time planning visits. These challenges include:

- Laying out multiple exposures to create a mosaic across a large target.
- Imaging a single target with a variety of instrument configurations.
- Planning not just the individual exposure times, but also the overhead time necessary to perform other tasks such as guide star acquisition, instrument overheads and telescope slew.
- Expressing a variety of constraints between exposures, such as "exposure B must run no later than one hour after exposure A".

These are currently manual, iterative processes that involve balancing exposure times to achieve the desired science objectives while keeping within the overall time constraints. The Exposure Planner will allow the user to express constraints and plan exposures visually using a graph with drag-and-drop capabilities. Behind the visual interface will be an expert system that understands the constraints imposed by the instrument and observatory, and can recommend an initial exposure plan. It will be able to recommend an optimal trade-off between individual exposure times and total visit execution time.

Development of the Exposure Planner has not yet begun. Its development will be a primary focus of the team in fiscal year 1999.

Re-validation Agent

This module has not yet been fully analyzed and development of the module will not begin for over a year. Currently, the Program Coordinators at the Institute spend a great deal of time re-processing already approved, but still pending proposals when a change to the instrument occurs. These changes can include a variety of things, for example, new calibration information that affects optimal exposure times. Currently, the concept for the re-validation agent is to use agent-based technology to evaluate the impact of changes to both submitted proposals and proposals that are still under development. The agent could seek out impacted proposals, calculate the effect of the impact, develop a recommendation and then notify the observer and observatory staff of possible alternatives.

Technologies

Since the focus of this project is software for more distant future use, we are intentionally trying to stretch existing software technologies and are aggressively using new and emerging components and development environments. To promote widespread usability in a community that uses several different operating systems, SEA is being developed in the latest version of Java and thoroughly takes advantage of the newest Java technologies (Java Foundation Classes, JavaBeans, Java 2D, Java Plug-in, and Object Serialization). We are also experimenting with Digital Signatures to avoid applet-related security restrictions and allow the SEA applet to provide the same functionality as a standalone application.

The SEA is designed to be easily extensible. This feature was necessary to accommodate both HST and NGST, but should be easily extended to other observatories and instruments. The SEA design is heavily object-oriented. The SEA leverages the dynamic nature of Java to allow extensibility without modifying existing code.

In order to integrate rule-based technology we are using Neuron Data's Advisor/J Java-based rules development environment. Advisor/J is a 100% Java rule-based system that includes a graphical rule editor and interactive debugging tools that assist us in developing and maintaining larger and more complex rule sets. Advisor/J also provides the ability to incorporate a rule-based engine in a fully distributed application. This will allow SEA to process rules on a user's local machine without constantly requiring server-based information or processing.

Conclusions

At this point, we have a functional prototype that is largely complete for single-exposure ACS proposals. While it is still too early to judge the effectiveness of the tool, we are strongly encouraged by the enthusiasm that the prototype has generated. Several groups have expressed interest in possibly adopting all or part of the SEA for their purposes, and the STScI has adopted our Exposure Time Calculator as the operational ETC for ACS. We are also encouraged by the ability of the still fairly

young Java technology to integrate and distribute applications using different technologies such as a rule-based engine with a fairly separate user interface system. Our primary challenge now is to ensure that we develop a rule-base that is both scientifically sound, comprehensive, and reliable enough to provide the observers with guidance so that they will neither need nor miss a large human support staff. Our secondary challenge is to stay in synch with developing software technology so that our development environment will remain current as the SEA's capabilities grow and mature.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 90-97

ISSN 0100-7386
Printed in Brazil

Automated Validation Process of User Interface Systems

Mourad Ould
Centre National d'Etudes Spatiales (CNES)
18, avenue Edouard Belin, 31401 Toulouse Cedex 4, France
ould@cnes.fr

Bruno D'ausbourg
Onera-Cert
2, avenue Edouard Belin, BP 4025, 31055 Toulouse, France
ausbourg@cert.fr

Abstract

The purpose of this paper is to present the research results on user interface systems (UIS) automatic validation, based on Lustre language formal specification. It describes the software prototype we have developed, and which allows to generate a formal model of the UIS and to automatically verify properties on it. It then proposes axes of optimization in order to validate this prototype on significant operational space ground segment UIS.
***Keywords**: User Interface System, Human Computer Interaction, Graphical User Interface, Formal models of interactive systems, Verification, Test, Lustre.*

Introduction

The importance of user interface systems (UIS) in data processing applications is steadily increasing. Space ground segments software do not escape the tendency : their UIS parts development costs approximately 30% of their total development effort and surely more concerning the validation and qualification aspects. In front of such a cost, and if we consider the increasingly complexity of the UIS part on those ground segments applications, it is necessary to think about the means of optimizing their validation.

As the UIS part ensure the well understanding of the internal state of the application, it must behave as intended. This point is particularly important when those interactive systems drive critical applications such as "Payload telecommand" or "Telemetry surveillance" within space ground segments.

The UIS behaviour constraints have to be taken into account at the first stage of the application development, i.e. the specification, in order to get a final system in conformity with them.

We can distinguish two parts in the UIS : the static part and the dynamic one. The constraints related to the static part of the UIS can be, for instance :

- The minimum width size of a window must be greater than X pixels,
- The maximum number of colors used in the same window must be lower than Y,
- The minimum font size of characters must be greater than Z points.

Those concerning the dynamic part can be, for instance :

- Interface reactivity : the interface must react and provide a feedback to each user action,
- Interface conformity : the interface must reflect every change of the application internal state,
- Interface robustness : the user can never be blocked (no deadlock),
- Screens alternation or mutual exclusion : screens X or Y are always displayed before screen Z, which in turn never appears when screen T is displayed,
- It Takes always less than n actions to perform to go from screen X to screen Z,
- There are always less than n screens displayed at the same time.

Most of the constraints related to the static part of the UIS can be verified quite simply by syntactic analysis of the code. This paper focuses more particularly on the dynamic aspects which are more complex to process.

The problem is to define properly these constraints during the specification process. In practice, there is no really such a specification process for UIS as it is the case for classical software systems. The used specifications are rather mock-ups or prototypes of the UIS than usual written specifications. Therefore, it becomes essential to verify exhaustively that these mock-ups and prototypes describe a system that behaves really in accordance with these requirements.

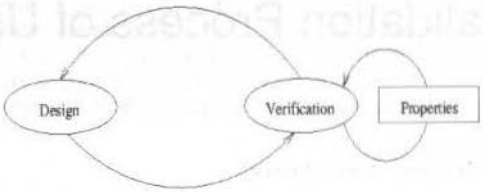


Fig. 1 An iterative design process.

Techniques for automated verification are very useful in this context. These techniques are making an explicit use of formal languages. These languages permit to describe and to operate computations on UIS formal models. But the problem is that the UIS designer has to be familiar with the formal technique. To bypass this problem, our approach proposes to build automatically the formal model of the UIS from its description as produced by an interface generator. This approach has several goals :

- to produce formal models that reflect exactly, by construction, what was devised by the designer;
- to hide formalisms to the designer who uses only the interface generator;
- to compute formal models quickly and as many times as needed.

In this case, producing a UIS formal description is helpful in order to provide developers with a verified mock-up or prototype. This formal description can be used also as a reference for evaluating the final system once it is developed. Indeed, having a formal model and a formal statement of user interface constraints at one's disposal allows to build test cases that permit to compare the final product with its requirements. So this leads to achieve a development process as depicted in Fig. 2.

In this case, producing a UIS formal description is helpful in order to provide developers with a verified mock-up or prototype. This formal description can be used also as a reference for evaluating the final system once it is developed. Indeed, having a formal model and a formal statement of user interface constraints at one's disposal allows to build test cases that permit to compare the final product with its requirements. So this leads to achieve a development process as depicted in Fig. 2.

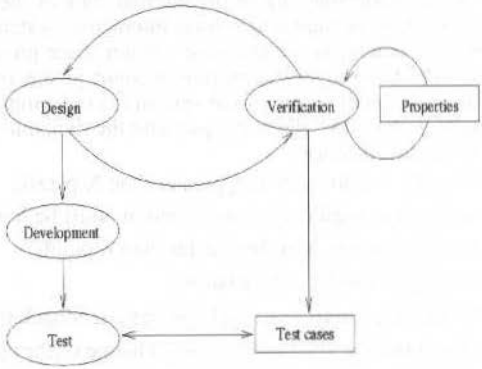


Fig. 2 The whole development process.

The paper describes on which formal basis this approach is founded and describes also how it is implemented inside the validation environment prototype. Section 2 describes a general model of interactor (i.e. any graphical object allowing the user to interact with the system). Section 3 gives some basis on the formal language Lustre used to denote models of UIS. Section 4 describes the prototyped environment and the main tools necessary to implement the suggested approach.

A Generic Interactor Model

If we consider the state of art, authors of works on UIS formal modelisation conclude that it's necessary to have at one's disposal a formal language to describe events and states. We do believe that the dichotomy between event and status phenomena can be taken into account by describing interactor as a dataflow system.

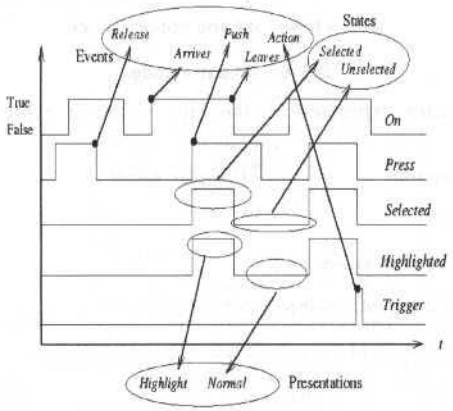


Fig. 4 Boolean flows of a pushButton interactor

Informally, a flow is a sequence of values. Fig. 4 depicts boolean flows that describe the behaviour of a pushbutton interactor. Events can be depicted by leading (*Push, Arrives)* or trailing (*Release, Leaves)* edges of flows, while states refers to measurable values that persist (*Highlight, Unselected*).

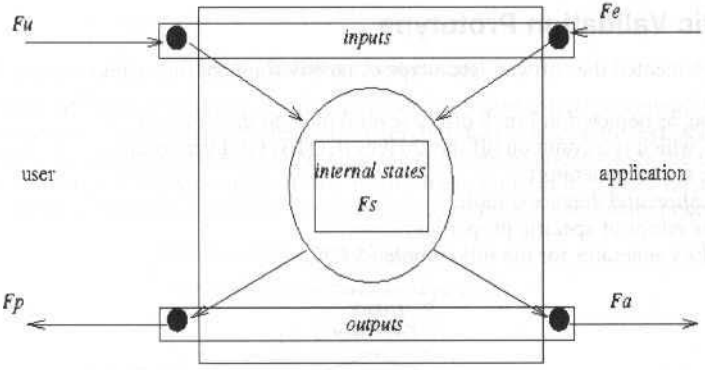


Fig. 5 Generic structure of an interactor.

Figure 5 describes the structure of an interactor as a set of boolean flows:

$F = Fu \cup Fe \cup Fs \cup Fp \cup Fa$

where each subset of flows can be defined by:

- - *Fu* : user actions events ;
- - *Fe* : application issued events ;
- - *Fs* : interactor internal states ;
- - *Fp* : interface presentation states/events ;
- - *Fa* : actions on the application triggered by the interactor.

Why Lustre ?

Lustre language is particularly adapted to express, in a formal dataflow notation, UIS interactors as modelled in Fig. 5.

A Lustre text is a set of declarations expressed in an unordered list of equations. An equation $X=E$ where E is a Lustre expression of flows specifies that the flow variable X is always equal to the value of flow denoted by E.

```
node edge(X:bool) returns(E:bool);

let E = false →(X and not preX); tel
```

Fig. 6 A Lustre node.

The structure of an interactor expressed on the Fig. 5 can be expressed in Lustre by a node as depicted in Fig. 7.

```
node interacteur(                E_display, E-hide, ...
UDS_On,
UAG_Push, UAG_Release,...)
returns(P_displayed,P_change_pres,P_xxx,....,
S_change_state, S_yyy,....,
A_ppp,...);
var                              -- declaration of local internal flows
let                              -- equations of flows
tel
```

Fig. 7 Structure of an interactor in Lustre

An Automatic Validation Prototype

We have implemented the software prototype of an environment that achieves some formal validation operations.

This prototype, as depicted in Fig. 8 offers several tools to the designer:

- - UIM/X, which is a common off the shelves (COTS) GUI generator ;
- - a Lustre model generator;
- - a verification and diagnosis tool;
- - a graphic editor of specific properties;
- - a test cases generator for the full completed UIS.

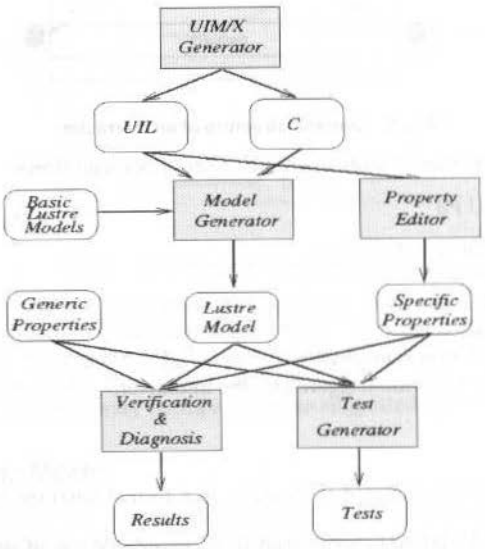


Fig. 8 An automatic validation prototype.

Generating a Lustre Formal Model

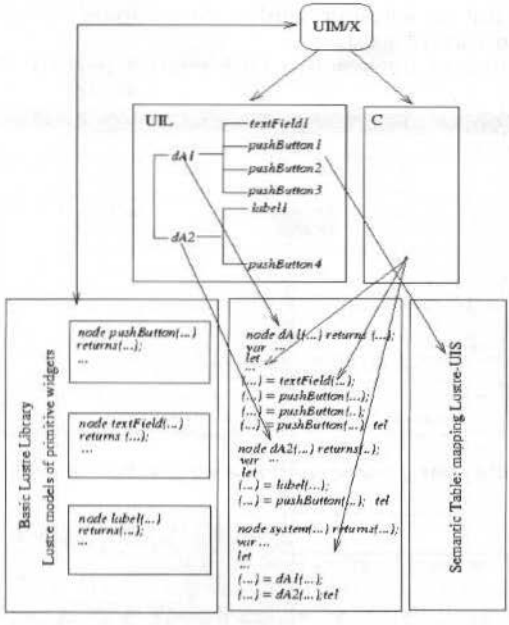


Fig. 9 Building a Lustre model

Figure 9 depicts the process of building the formal model of a user interface system that was devised with the UIM/X generator. An analysis of the UIL (User Interface Language) text, that describes the hierarchy of the involved interactors, permits to produce the structure of nodes inside the text in Lustre.

Each primitive widget in UIM/X is mapped to a Lustre basic node in the Basic Lustre Library. The representation of composite widgets is obtained by composition of these basic Lustre nodes.

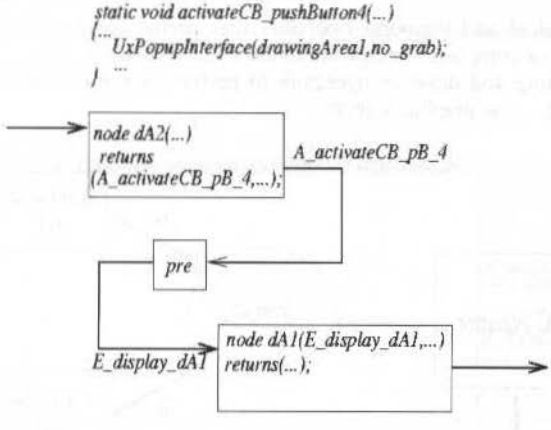


Fig. 10 Data flow analysis and interconnection of Lustre flows

The analysers that permit the automatic generation of these Lustre models were developed within the ONERA-CERT *Centaur* system.

Properties Formal Verification

Lustre is also a temporal logic notation and allows to perform verifications by model-checking. For instance, the *conformity* of an interactor can be defined by the following expression, where the term *implies* is the logical implication operator:

implies (S_change_state_*, P_change_pres_*)

This expression states that an interactor satisfies the conformity property if its presentation is modified when its internal state has changed.

Figure 11 gives an outline of the prototype GUI during a property selection before its formal verification.

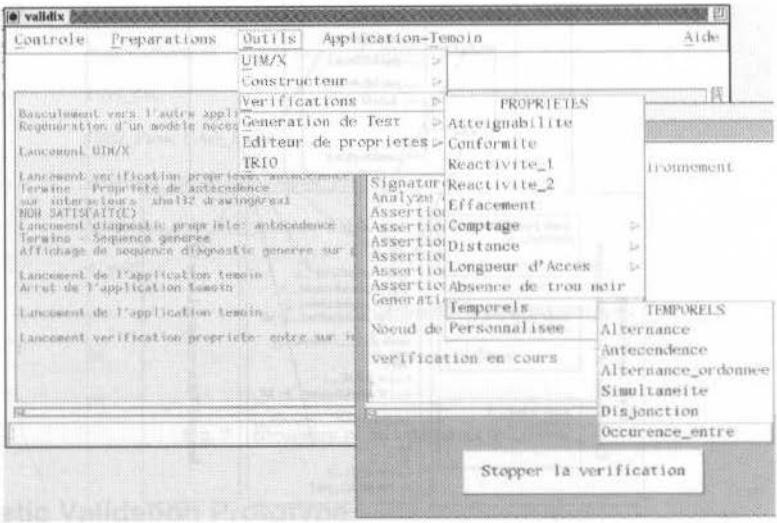


Fig. 11 Selecting a property using the prototype.

Graphic Editor for Specific Properties

This property editor allows the user to build schemes of logical and temporal relations between the interaction objects that are involved in the UIS. It offers a set of palettes to the user :

- - a palette that contains all the types of interactors that are used in the UIS and captured by the formal model ;
- - a palette of logical and temporal operators that permit to draw graphically some relations between the basic or composite interactors in the UIS ;
- - a palette of editing and drawing operators to perform some classical editing operations (cut, paste, copy, delete,...) on graphic objects.

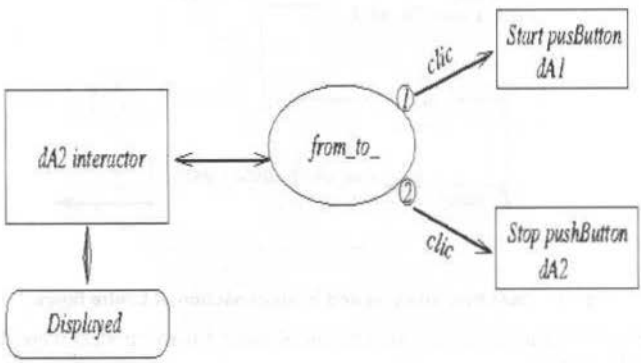


Fig. 12 Graphical expression of a temporal property.

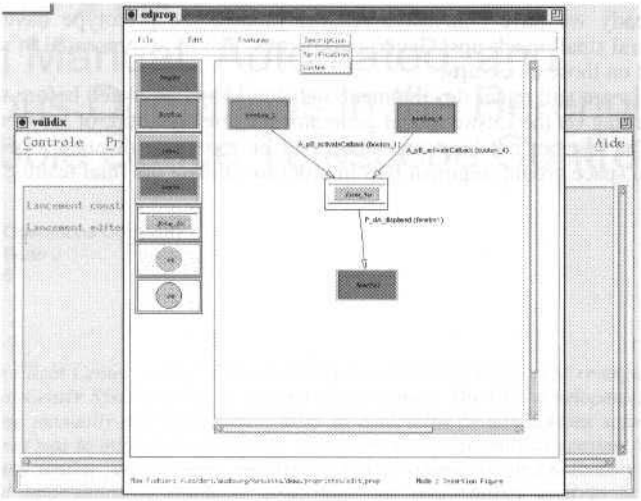


Fig. 13 An outline of the specific properties graphic editor.

Generating UIS Test Sequences

When the verification process succeeded for a property P onto an interaction structure I, the environment suggests to build an exhaustive set of behavioural traces that lead to particular configurations of the UIS that satisfy P. These traces can be viewed as test sequences that will be used later in order to check that the final interactive system, once developed, satisfies P on I. Generating tests is then a welcome side effect of verification and diagnosis operations and is also based on observers. The technique used to produce the test observer in Lustre are based on instantiating generic properties and inserting some relevant Lustre expressions associated with the tested properties.

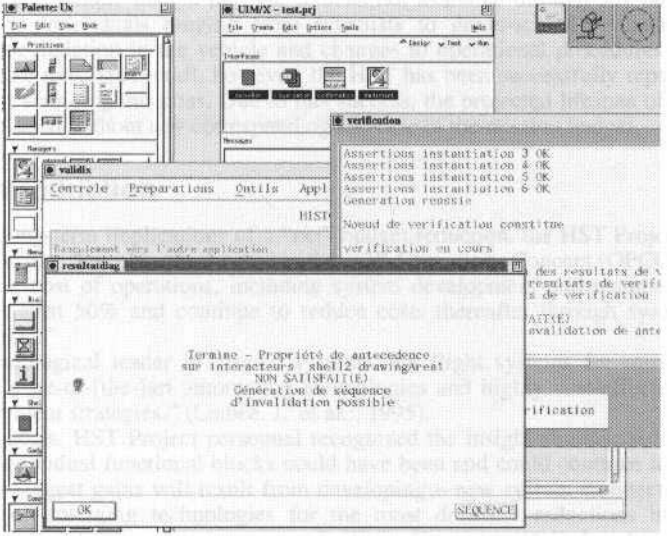


Fig. 14 An outline of the prototype diagnosis tool.

Conclusion

Among the advantages of the approach described in this paper, an important one is the fact that formalism is hidden to the designer ; the latter can thus focus on the UIS development, using any GUI generator producing UIL code.

Of now and already we have conclusive results. Indeed, the prototype have been successfully experimented on several small mock-ups. Generic properties, like those presented in section 1, have been automatically verified on those mock-ups.

The research works are still under development and should be completed before the end of this year. Currently, we are working on the Lustre model generator parameterization, of the in order to avoid huge models generation. The last but not least main step will be the experimentation of the prototype on a significant operational space ground segment UIS in order to validate the final results.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 98-104

ISSN 0100-7386
Printed in Brazil

Balancing Manual, Automated, and Autonomous Operations in the Re-engineering of the Hubble Space Telescope Control Center

David G. Fish
Lockheed Martin Space Operations Company
9701 Philadelphia Way, Suite J
Lanham, Maryland 20706
david.fish@lmco.com

Abstract

NASA's Goddard Space Flight Center (GSFC) instituted the Vision 2000 (V2K) effort to re-engineer the Hubble Space Telescope (HST) Control Center System (CCS) by the end of the century. The CCS development team investigated the feasibility of reallocating manually-executed, human tasks to automated or autonomous software components. The purpose of the assessment was to minimize operational costs without significantly increasing risk. During this effort, we found that while many similar projects extol the virtues of both automation and autonomy within control centers and spacecraft, limited information is available concerning the methodology used to achieve an appropriate architectural balance.

This paper introduces the V2K project and its objectives, defines manual, automated, and autonomous operations, and describes the approach the CCS team used to allocate operational tasks within the comprehensive system. The goal of the process outlined is to optimize the distribution of tasks among users, automated processes, and autonomous functions while satisfying mission objectives and risk thresholds. This paper concludes with a progress summary and lessons learned.

Keywords: *Automation Processes, Cost-Effective System Design, Methodology*

Project Context

The Hubble Space Telescope (HST) is recognized throughout the world for its scientific contributions. It enables unique insight into phenomena not available through any other method and is, therefore, familiar to individuals ranging from scientists to grade-school students. Like any other spacecraft, gradual degradation in the vehicle and changes to operational procedures have impacted its ground system. Unlike most spacecraft, however, the HST has been successfully repaired and upgraded through two on-orbit servicing missions. Due to this success, the projected lifespan of the HST has been extended to the year 2010 without any corresponding increase in the mission budget.

Vision 2000 Introduction

To address the long-term implications of a "net" budget reduction, the HST Project has initiated the Vision 2000 (V2K) effort. The objectives listed in the V2K Operations Concept (OPCON) include:

1) "Reduce the cost of operations, including system development and maintenance, by the year 2000 by at least 50% and continue to reduce costs thereafter through system and efficiency improvements."
2) "Be a technological leader in ground systems and flight systems for mission operations by employing state-of-[the-]art automation technologies and highly cost-effective implementation and management strategies." (Leibee, J. et al.:, 1995).

To attain these goals, HST Project personnel recognized the insight summarized best by Zillig and Benjamin: "While individual functional blocks could have been and could continue to be replaced ..., it is expected that the biggest gains will result from developing a new system architecture that makes the most efficient use of emerging technologies for the most dramatic reductions in size, operational complexity, and operational and maintenance costs." (Zillig, D. et al.:, 1994). In other words, the 15-year old legacy ground systems with their functional overlap were not worth patching any longer.

It is important to note that V2K is neither attempting to alter the GSFC or NASA-provided institutional services, nor placing control of the payload in the hands of principal investigators, nor pushing for "lights-out" operations, nor striving for full spacecraft autonomy. V2K is about sensibly re-engineering the operational tasks and ground system elements associated with Planning and Scheduling (P&S), Flight Software (FSW), and the Control Center System (CCS). The remainder of this paper will focus on the CCS.

CCS Tenants

The following basic principles were identified to guide the CCS development effort. First, the development process would have to be streamlined and system capabilities would be incrementally delivered in order to complete the ground system re-engineering by the year 2000. Second, Commercial Off-The-Shelf (COTS) and Government Off-The-Shelf (GOTS) products would be used wherever practical and all newly-developed software would be written in C++ or Java. Finally, web-based technologies would be provided to unite the global HST community through a platform independent user interface.

CCS Stakeholders

The CCS team includes representatives of HST Project Management, the Flight Operations Team (FOT), HST Subsystem Engineers, System, Software, Test, and Network Engineers, the science community, and instrument developers. Experts in each of these areas were actively recruited and necessary training was provided to complement existing skills.

This group was then supplemented with specialists of varying disciplines: Object-Oriented Analysis and Design, COTS Product Support, Security, Performance Analysis, Human Factors, and Knowledge Engineering.

To facilitate development, the individuals sacrificed corporate identities and donned CCS badges when moving to the co-location (COLO) facility. By locating the core development team in a single building, including anticipated users, (Land, S. et al.:, 1995), and minimizing management bureaucracy, the team was equipped and empowered to meet the challange.

CCS Status

To date, CCS encompasses over 600K new lines of code (LoC) and 300K LoC of ported legacy[1] software. Over 25 COTS and GOTS products have been integrated and the architecture includes SGI, HP, Sun, and WinNT/95 platforms. The software currently runs in parallel with the legacy system so that the FOT can compare CCS results with those observed in a known baseline. This approach also provides a backup system in case problems arise. CCS is planned to be used as the primary operational system starting in February 1999 and will be used to fly Servicing Mission 3 in mid-2000. It now includes most of the functionality required to support manual operations. In addition, a concerted effort was made to reduce or eliminate operator involvement in routine tasks by including many automated and autonomous functions. Examples of current automated or autonomous capabilities include: telemetry processing (receive, process, route, merge, store, and retrieve), CCS initialization and status management, disk space management, and Discrepancy Report (DR) / Change Request (CR) management and distribution.

Methodology

The purpose of this paper, though, is not to describe the CCS architecture or to espouse the virtues of integrating emerging technologies. The intent is to summarize the methodology used to allocate operational tasks to manual, automated, or autonomous modes of task execution. The CCS approach is based on the premise that the objective of any development effort is to deliver a system that will 1) accomplish the operational objectives and 2) do so in the most cost-effective manner, (Hoban, F. et al.:, 1995). This requires a thorough understanding of the tasks to be accomplished, available technologies, associated risks, and accessible resources.

Systems Engineering Context

Traditionally, information processing systems have been viewed as consisting of three components: Hardware (HW), Software (SW), and Users. In order for any system to achieve its stated objectives, each of these three elements must play the appropriate role. In fact, during system analysis and design, a comprehensive set of operational tasks must be defined; each task is then individually allocated to one of the three components. These tasks eventually "drive out" functional and performance requirements. If the

[1] Due to the projected magnitude of a sufficient test and validation effort, legacy software from the Command and Sensor Analysis subsystems has been ported from VMS FORTRAN to the UNIX platform.

design appropriately allocates each requirement and if each system element is able to perform all of its assigned duties, then the overall system is "competent."

In re-engineering the HST ground system, however, minimal customized hardware was required. Therefore, the development team viewed the hardware, network, and COTS / GOTS software as comprising the system infrastructure. Tasks were then allocated to the infrastructure, newly developed custom application software, or the user (Figure 1).

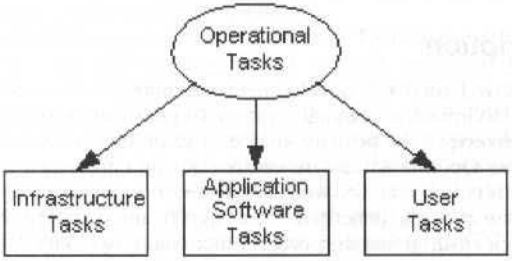


Fig. 1 Allocation of Operational Tasks.

While most development efforts strive to establish a competent system, the expense of human intervention required over the life of the mission may become cost-prohibitive. This raises the second objective of minimizing program costs, in other words, the cost-effectiveness of task completion. This concept is important because it encompasses the notion of balancing development costs with on-going operational expenses, that is, maximizing the return on investment. The project development team must ensure that the selected approach satisfies the functional and performance requirements while making the best use of available resources.

rom this perspective, tasks may be labor-intensive manual processes, user- or system-initiated automated procedures, or autonomously executing intelligent agents. Discrete boundaries, however, do not exist between the "modes" of operation; they are logically viewed on a continuum (Figure 2.) Movement of an individual task toward autonomy happens as human intervention is eliminated from the task or as human decision making is incorporated within the infrastructure or application software.

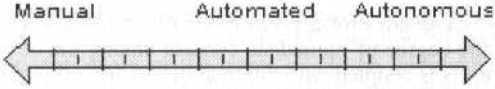


Fig. 2 Operational Mode Continuum.

Yet, any given system does not exist at a single point on the continuum. Instead, each operational task (or thread) should be implemented in a mode that provides the most cost-effective, acceptable risk approach. In addition, most systems will allow multiple modes of executing a small subset of the operational tasks. Typically, one mode is entirely manual and the other mode includes some degree of automation or autonomy.

Although some autonomous functions utilize Artificial Intelligence (AI) technologies, automation and autonomy are not synonymous with AI. Many completely autonomous tasks, for example, automated backup procedures, contain virtually no simulated intelligence. Similarly, some manual tasks, like realtime commanding done to recover from an anomaly, can require extensive human intelligence. The relative degree of intelligence incorporated into automated or autonomous task completion is a separate dimension beyond the scope of this paper.

Finally, automation and autonomy are not end goals. In fact, the use of automation or autonomy may not reduce operational or lifecycle costs, (Carraway, J. et al.:, 1996). Rather, automation and autonomy are candidate approaches for attaining efficient system designs that need to be seriously considered—their "importance ... cannot be overestimated." (Gamble, E. et al.:, 1998).

Competency and Mode Coupling

During a system's lifecycle, new operational tasks will be defined. This usually happens as new requirements are levied on the system, hardware components fail, or critical software bugs are discovered. Typically, the users are allocated the responsibility of executing the new tasks required to maintain overall system competence. Periodically, though, tasks are reallocated and system changes are required in the infrastructure or application software components. When this re-engineering occurs, the responsible developers must properly consider competency (i.e., ensuring the tasks can be performed) and operational mode (i.e., ensuring cost-effectiveness).

The following example illustrates this relationship. In legacy HST operations, the task of recording history data was performed by users mounting 9-track tapes on the respective drive. When the task was re-engineered during CCS development, a robotic tape juke box was installed with software to manage the tape volumes. System competence was maintained by reallocating the user's task to software components. In the meantime, by moving to an automated mode of operations, the cost-effectiveness of the system was increased.

Methodology Description

The methodology employed for CCS development attempted to balance these two issues. At the "macro" level, the process followed a nominal System Engineering approach for a large, distributed system with incremental deliveries. The primary source of requirements was the existing system and its corresponding documentation. Operational task re-engineering and prototyping were used to identify new or modified requirements. External reviews were conducted on a regular basis to provide mid-course corrections. Additionally, the process benefited from significant participation by the eventual users. Overall, this general approach ensured that the correct functionality was developed.

"Micro" Process Steps

Within the larger context of the macro process, there were smaller, embedded processes utilized to develop CCS. It is primarily these micro process steps that ensured cost-effective operations by allocating tasks to manual, automated, or autonomous system modes. Ignoring the remainder of the development process, the two steps of interest are 1) the Initial Mode Assessment and 2) the Mode Reassessment. The primary difference between the two steps rests in the focus of the activity. The Initial Mode Assessment concentrated on allocating the "obvious" tasks correctly. This step comprised most of the up-front analysis activity performed for CCS development. The Mode Reassessment focused on allocating the "obscure" tasks correctly. It began after a year into the development effort and continues today. The key to realizing an appropriate balance during system mode allocation is differentiating the obvious from the obscure tasks.

Defining the "Obvious"

Some tasks are "obvious" candidates for a given mode of operations. Tasks whose mode is obvious are generally recognized by the degree of immediate consent among the development team. Essentially, the concept proposed by the users is adopted with minimal discussion or modification.

For example, due to the routine nature of receiving realtime telemetry data and subsequent recorder dumps, the process of merging both data versions to create a single contiguous data stream and loading the result into the data warehouse is nearly autonomous. Users can manually override the process when required, but the vast majority of the data is processed without human intervention. As a second example, since servicing missions only happen at three-year intervals and involve risks to humans in spaceflight, all procedures are obviously executed in a manual mode since the cost of automating the process would be higher than any potential cost savings.

Identifying the "Obscure"

By definition, "obscure" tasks are "not easily understood or obviously intelligible." They are "vague" or "ambiguous." (McKechnie, J. et al.:, 1993). For example, during nominal operations, realtime commanding is used for less than 5% of HST's commands; it requires a very small operational staff with significant contextual knowledge. These factors push the task toward a manual mode of operation; however, automating the activity could reduce the likelihood of user-induced errors—especially those having serious consequences.

Operational tasks where the mode is not easily discernable can be identified by disagreement among team members or passionate technical debates. Frequently, discussions will consider the merits of automation or autonomy. Unlike the "obvious" tasks whose mode is normally driven by the user, the mode determination for obscure tasks must be resolved by engineering analysis. However, once the analysis is complete and the nature of the task is better understood, the implementation mode usually becomes more "obvious" to the developers.

Initial Mode Assessment

As indicated earlier, the initial mode assessment started shortly after CCS development began. After the Methodology Team determined the process and project standards, the Operations Concept and supporting Scenario Development activities began. Subteams were formed based on operational

boundaries and each group was responsible to address all key operational tasks. Multiple experts from each relevant domain were included. This iterative effort continued for several months—until sufficient detail was provided for several levels of scenarios and corresponding system capability descriptions. This job re-engineering effort ensured that non value-added steps were eliminated instead of being incorporated into the CCS, (Lyon, J., 1996). The methodology recognized the need to "capture an understanding of a domain and its tasks rather than the behavior of its experts; task analyses must provide the first phase of system design.", (Abramson, B., 1990).

Once this phase was well underway, the Proof-of-Concept Team was established to prototype sample tasks. The effort intended to validate the operational concept, technical feasibility and supporting requirements. Weekly progress demonstrations were provided to the CCS team and HST Project. In parallel, the Architecture Development Team identified external interfaces, defined CCS subsystems, and translated the operational scenarios into thread descriptions and corresponding diagrams. This activity verified allocation to system components and operational modes for the "obvious" tasks.

Two additional efforts originated in this period. First, the CCS Architecture Board (CAB) was chartered to prioritize development activities and to oversee the specification of interface descriptions. Second, the actual software development teams began evaluating COTS products and prototyping application software. The prototyping activities reduced risks by identifying requirements and demonstrating feasibility. Each software team collaborated with operations personnel to establish detailed requirements, resolve design issues, and finalize the mode allocations.

Overall, the first micro step of interest lasted approximately six months. The major focus of this activity was to identify the preliminary component and mode allocations for each operational task. The result was a thorough understanding of how the primary system functions would work; some were manual, some were automated, and a few were relatively autonomous. In addition, the preliminary specification identified how some of the more "mode-obscure" tasks would be implemented.

Mode Reassessment

The second micro process step began approximately 1 ½ years after the initial mode assessment. By then, the majority of the CCS infrastructure was functional. The software development teams for each CCS subsystem had tailored their respective approaches; (Ayache, S., et al.:, 1994), yet, each group adhered to project-established standards. This freedom, and a high productivity rate, had provided 400K LoC through five delivery cycles. The independent GSFC software evaluators rated the software extremely high and several related projects were already investigating the reuse of CCS software.

At this time, progress had been made in all modes of operation and development of some of the mode-obscure tasks was imminent. More of the teams would be integrating emerging technologies or other advanced features. Recognizing that the risk factors in developing systems utilizing expert systems technology must be assessed continually, (Gilstrap, L., 1991), the team responded proactively.

The Automation Working Group (AWG) was established with the responsibility of verifying the anticipated mode of operation for key tasks and system functions. In this micro process step, subgroups of the AWG used multiple, independent approaches to validate the current mode allocations.

- *Approach 1—"Do it again."* Composed primarily of operations personnel, the first validation team used metrics[2] to prioritize key tasks and quantify the anticipated cost-effectiveness of increasing task execution autonomy. The final products included a comprehensive concept for autonomous HST operations, (Bradley, A., et al.:, 1998). It included graphical and textual system descriptions with corresponding requirements. This product served as a "benchmark" for several other activities.
- *Approach 2—"Find the 'tall poles.'"* In this approach, large pockets of existing staff were identified. An informal task analysis was performed to identify operational responsibilities; results indicated that the assigned duties were already sufficiently incorporated in the CCS design.
- *Approach 3—"Count the beans."* A large matrix was constructed identifying operational tasks and subtasks. Metrics for technical feasibility, operational and science risks, and lifecycle costs were estimated. Personnel also reviewed operational logs to validate estimates and to search for missing tasks. Although the overlap between subtasks provided some corroboration, the limited usefulness of the product did not seem to justify the time required to capture the data. Therefore, the matrix was never fully populated.

---

[2] Although the AWG identified metrics for CCS, Appendix B of the ESDM Standard provides a comprehensive list of candidates (see the endnote for Gilstrap for a reference).

- *Approach 4—"Read the directions."* This approach obtained copies of existing procedure documents for routine and contingency operations, as well as fault isolation procedures. The procedures were reviewed to identify the current task completion approach and assess the feasibility of increasing autonomy. The effort was not very productive since CCS versions of the procedures were incomplete and the process re-engineering efforts had changed the way operational tasks would be performed in the new system.
- *Approach 5—"Return to the familiar."* This approach depended on reviewing the existing CCS architecture and ensuring all tasks were allocated properly. In addition, pending CRs and HST Anomaly Reports (HSTARS) were examined to isolate impacts. The functionality remaining for each CCS subsystem was classified, prioritized, and scheduled for development. Risks were assessed and the need for knowledge bases identified.
- *Approach 6—"Two heads are better than one."* Throughout the CCS lifecycle, collaboration with other similar projects was cultivated, vendors contacted, demonstrations received, and literature read. This approach clearly added value on multiple occasions.

Lessons Learned

While some approaches were more effective than others, each of the reassessment approaches provided insight on the appropriate allocation of tasks to operational modes. Throughout the activity, confidence in the CCS design gradually increased. The minor discrepancies identified were integrated with the design documents and Release Integration Plan (RIP).

By conducting frequent internal and external reviews, we remained focused and avoided blindly automating the "old way" of performing a task. Errors of this type can be extremely costly and may reduce the overall competence of the system.

Changing engineering processes or COTS products in midstream can help despite the legitimate tendency to minimize "environmental" changes. As already stated, different software development teams were permitted to use alternate methodologies. This allowed local optimization but the project standards helped ensure overall quality and consistency. In addition, we held off-site process evaluation sessions to incorporate continual process improvement philosophies. During one review, we recognized that the audience and usage of the RIP had changed significantly since its inception. We developed a new format and updated the product accordingly. Likewise, our original plan to use Java applets running under Netscape on SGI workstations was changed when vendor support for the virtual machine lagged behind. Instead, we switched to compiled Java applications running on WinNT platforms. These changes, and several others, were not undertaken without thoughtful consideration. However, we addressed the issues directly and moved forward.

Similarly, COTS and GOTS products are not a panacea; don't wait too long for a "better" version or product. In many cases, it is better to continue forward momentum than to delay. For example, one development team used the web as a design and development repository. The lack of well-designed tools increased the production overhead, but developers still benefited by receiving an easily accessible data store.

Conclusion

On-going mode reassessment is inevitable and appropriate. It can happen at any time in a mission lifecycle as tasks are added, system components fail, or as in the case of HST, servicing missions deliver upgraded scientific instruments. Changes may arise during initial development of a new system or during the re-engineering or maintenance of an existing system. It is, therefore, imperative that development, operations, and maintenance personnel are equipped to perform this activity in a disciplined manner. Regardless of the specific methodology selected, the team is obligated to ensure operational system competence and overall lifecycle cost-effectiveness. The methodology described in this paper has been used successfully for the re-engineering of the Hubble Space Telescope Control Center System and may provide value to other projects attempting to balance the allocation of operational tasks.

References

Leibee, J. (Chairman), Barbehenn, G., Boyce, L., Carpenter, K., Douds, D., Doxsey, R., Hallock, L., Ochs, B., and Sobieski, S., February, 1995, "Vision 2000 Operations Concept".

Zillig, D. and Benjamin, T. Nov. 1994, "Advanced Ground Station Architecture," Proceedings from the Third International Symposium on Space Mission Operations and Ground Data Systems.

Land, S., Malin, J., Thronesberry, C., Schreckenghost, D., July, 1995, "Making Intelligent Systems Team Players-A Guide to Developing Intelligent Monitoring Systems", NASA Technical Memorandum 104807.

Hoban, F. and Hoffman, E. (Directors), June, 1995, "2.3 Objective of System Engineering," NASA Systems Engineering Handbook, Reference SP-6105.

Carraway, J. and Squibb, G., September, 1996, "Autonomy Metrics," Proceedings from the Fourth International Symposium on Space Mission Operations and Ground Data Systems.

Gamble, E. Jr. and Simmons, R., September/October , 1998, "The Impact of Autonomy Technology on Spacecraft Software Architecture: A Case Study," IEEE Intelligent Systems and Their Applications, Volume 13, Number 5.

McKechnie, J., (Editor)., 1993, "Webster's New Universal Unabridged Dictionary", Second Edition.

Lyon, J., October, 1996, "Summary of Day 1 Activities," Proceedings from the Autonomous "Lights Out" Operations Workshop: Operational Challenges and Promising Technologies, GSFC Publication.

Abramson, B., August, 1990, "Competent Systems: Effective, Efficient, Deliverable," Proceedings from the Sixth Annual Conference on Applications of Artificial Intelligence and CD-ROM in Education and Training.

Ayache, S., Haziza, M., and Cayrac, D., November, 1994, "Delivering Spacecraft Control Centers with Embedded Knowledge-Based Systems: the Methodology Issue," Proceedings from the Third International Symposium on Space Mission Operations and Ground Data Systems.

Gilstrap, L., September, 1991, 'Expert System Development Methodology Standard', Revision 2, GSFC.

Bradley, A., Chu, R., Power, H., Sobieski, S., April, 1998, "Concept for the Autonomous Operation of the Hubble Space Telescope," internal CCS publication.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 105-112

ISSN 0100-7386
Printed in Brazil

The SCOS 2000 System and it's Application for the Integral Mission Control System

Roger Patrick
Michela Alberti
Michael Schick
TERMA Elektronik GmbH
Brunnenweg 19, 64331 Weiterstadt, Germany
rmp@terma.com

Abstract

The SCOS 2000 System is the latest mission control infrastructure to be used by the European Space Agency at it's Operations Centre (ESOC) in Darmstadt, Germany. It is a follow-on from the earlier MSSS and SCOS-1 systems, but embracing newer technologies and standards. The system has proven itself as a viable MCS on a number of missions. In the past year, significant additional development work has been performed on the system to allow it to be able to address the needs of more complex missions as well as the simpler ones. This has culminated in the selection of the system for the ESA Integral Spacecraft - a complex scientific mission. This paper gives an overview of the architecture and functionality of the system, the technologies employed and it's use for the Integral Mission Control System.

Introduction

SCOS 2000 is the latest generation of spacecraft control system developed at the European Space Operation Center (ESOC) of the European Space Agency (ESA). Initially known as SCOS II, it was used for the first time as a telemetry retrieval and display system for the joint ESA/NASA science mission SOHO in late 1995, acting as a complement to the SOHO control system. In September 1997 it was used as a full-control system for the Launch and Early Orbit Phase (LEOP) of the meteorological satellite METEOSAT-7. From October 1997 it was used for the TEAMSAT mission which was comprised of two spacecraft launched on ARIANE 502 and for the still on-going mission of the Huygens probe carried on the Cassini spacecraft towards Titan.

After these successes, the system been further developed to better support more complex missions, and is now known as SCOS 2000. The system is today baselined for the LEOP phase for the Eumetsat Meteosat Second Generation Spacecraft (MSG-1 and

MSG-2), the PROBA mission, the SMART-1 Mission and has recently been selected for the ESA Integral Science Mission. SCOS 2000 is today developed for ESOC by a consortium of companies led by TERMA Elektronik GmbH (Germany), including Vitrociset SpA (Italy) and SSL (UK). CAM GmbH (Germany) is also part of the consortium for the Integral Science Mission.

SCOS 2000 Architecture

The SCOS 2000 architecture is typical of that found in many distributed systems. It is comprised of a network of workstations, using a client-server paradigm, in which server processes provide services to the clients as a whole (e.g. database server, archive server). The system is scaleable in order to suit different missions requirements, budgets and/or mission phases. Low cost missions such as TEAMSAT used a single workstation for each spacecraft ("SCOS-in-a-box", Ref 1) whereas the METEOSAT-7 used a system of 4 servers and 22 client workstations (Ref.2) during the complex and critical LEOP phase of the mission.

SCOS 2000 provides the following key functionality, possible to be customised by client missions:

- archiving, distribution and retrieval facilities
- telemetry extraction, processing and display (e.g. alphanumeric, graphic, scrolling, schematic animations (mimics) and variable packet displays (PUS)
- commanding including multiplexing, on-board queue model management, command history, auto stacks etc.
- Operations language (e.g. OL parser/ interpreter and compiler)
- event and action handling (e.g. page support personnel, send command etc.)
- roles & privileges of operations staff
- autonomous file transfer system driven by database configuration
- homogenous high level man-machine interface
- on-board software management.

- ground station interfaces

The high level design of the SCOS-2000 system is shown in Fig. 1.

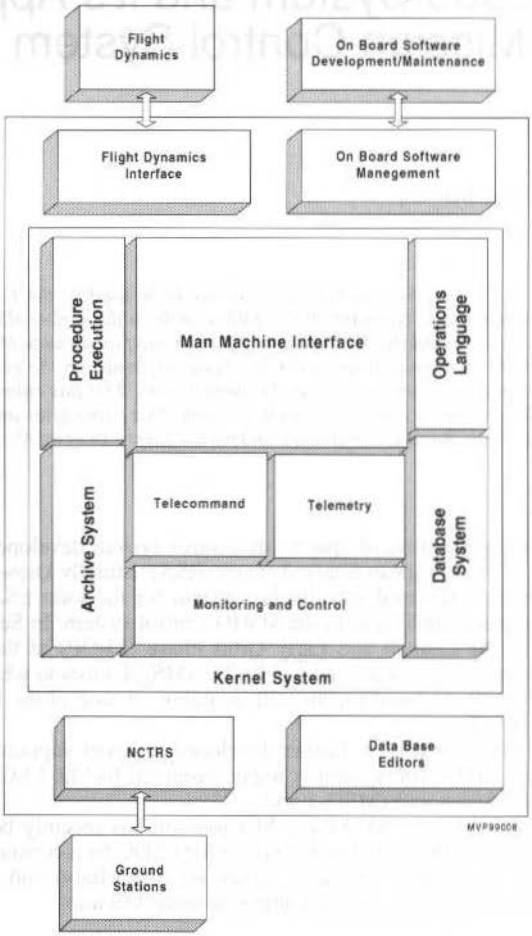


Fig. 1 SCOS 2000 – General Architecture

This shows the kernel of the system augmented with the Ground Station Interfaces (NCTRS) and the Flight Dynamics interface. The Database System with the appropriate set of database editors is left up to the choice of client missions – several possibilities exist (ORACLE, MS Access etc).

Figure 2 shows a layered view of SCOS 2000.

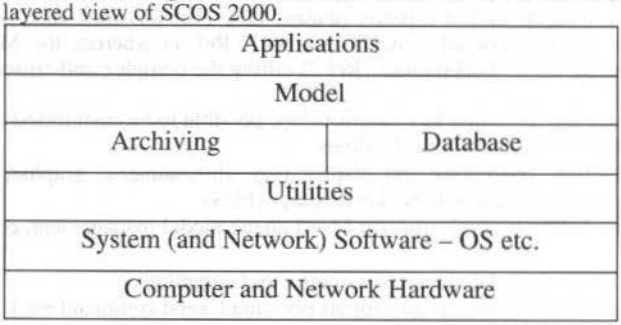


Fig. 2 SCOS 2000 Layers

The lowest layers are the hardware and system software. Next is that of the utilities, which include general services such as for Graphical User Interface (GUI) elements, time and date services and

communications. The database and the archiving layer make use of services provided by the utility layer. The model layer uses archiving and database functionality as well as services provided by the utilities. The highest layer comprises the applications, of which telemetry and telecommand displays are typical examples. Mission-dedicated software can use components of any layer. The system allows the addition of other applications to extend the functionality or specific applications for particular missions.

There is a great deal of flexibility in the specific network architecture that can be set up for a given mission. For example the kernel component can be located on one server, or duplicated on two servers or installed on several client workstations. The actual network design will depend on considerations of load sharing, reliability, and number of users. The client applications can run on any workstation including the server workstation.

The SCOS 2000 software makes use of Corba and UNIX sockets for communications between processes. This loosens the coupling between processes, so that changes can easily be made to the process architecture, allowing individual processes to be changed or upgraded. Only the clearly defined interfaces between the processes must be maintained.

Some of the key processes in the software architecture are highlighted below.

Archive System:

The archive system is responsible for archiving all data and events. It performs a number of other functions:

- **Packet Distribution Server** - Central to the whole system is the Packet Distribution Service. This application is operated on the server workstation receiving any kind of packets (telemetry, telecommand and event packets), stores them, and distributes them to applications via the Caches executed on the client workstation.
- **History File Retrieval** - the history file retrieval is responsible for handling all packet retrieval requests from client applications (via the Caches from the client workstations).
- **Caches** - the Cache tasks provide the interface between applications and the archive subsystem. Data received by the caches executed on client workstations (either live data or retrieved data) is stored in temporary caches, in order to reduce the load on the server. Separate caches exist for telemetry, telecommand and event packets.

Telemetry Desktop:

This is the main telemetry monitoring application, providing the user with a sophisticated tool for displaying telemetry in alphanumeric, graphical, scrolling and mimic form. Although providing backwards compatibility to the older systems using the same alphanumeric, scrolling and graphics layouts as the MSSS and SCOS-1. Enhancements have been included such as the Telemetry Query Display, which allows the user to examine all pertinent information relating to a specific parameter, via a point-and-click interface. Local changes to the alphanumeric displays are possible using drag-and-drop techniques. Calibration curves, limits and even the demotion of synthetic parameters may be changed online, allowing instant validation of changes. A user with 'Database Editor' privilege can also broadcast local changes to other client workstations.

Manual Stack

This task provides the user with a powerful utility for the preparation, modification and dispatching of telecommands. The user can prepare telecommands and/ or sequences for dispatch. Editing facilities such as cut, copy and paste are provided, for easy ordering of telecommands. Any PUS telecommands or sequences of commands can be loaded, and may also be saved again after modification for later reuse. Grouping, blocking and a lot of other powerfull mechanism are provided to the user for the management of commands. Inside the manual stack is also a Telemetry Query and On.board model queue display included.

Behavior checker:

The Behavior Checker performs the centralised telemetry checking, that is limit, status and status consistency checking. Alarms are generated in the event of violations, and state changes are sent for filing and display on out-of-limit displays.

Technologies employed

SCOS 2000 is an object-oriented spacecraft control system framework, with all the advantages that OO provides such as encapsulation, specialisation etc. The implementation language of the system is C++. It is scaleable - that is, it can be adapted to support different types of missions, ranging from small missions to complex ones.

The technologies are chosen to allow SCOS 2000 to provide a cost effective solution for spacecraft control. The main reasons are:

a) Customization is achieved without programming by means of database and configuration files. For a simple mission such as TEAMSAT (Ref 1), a basic system can be configured in less than two weeks with the help of operations staff performing the DB configuration.
b) Interfaces with other software or special equipment exploit an open-system design, client server architecture and use widely adopted standards, such as Unix System V Release 4, X11, POSIX, TCP/IP, CORBA and STL. With this approach there is no need to change the mission control system core design - the adaptation is normally achieved by plugging new components into the system.
c) Encapsulation hides the internal complexity of the implementation, simplifying the task of extending the system. This does not mean that extending a mission control system is a trivial task – a knowledge of the framework architecture is needed - but this feature makes it easier.
d) A number of COTS products are used in the system, where the cost/benefit analysis shows them to bring a significant advantage

Standards

The SCOS 2000 was designed to support CCSDS Packet Telemetry and Telecommand standards and so also readily supports older Telemetry and Telecommand standards. Some of the more recent development work on the system has been to implement services as defined by the ESA Packet Utilisation Standard - many of which are required for complex missions such as the ESA Integral Mission.

Basic Functionality

The following paragraphs are a brief summary of the main functions offered by the system.
Telemetry:

- Telemetry reception, extraction, processing and distribution. SCOS 2000 is designed to support CCSDS Packet Standards. Backward compatibility for format based spacecraft, (such as MTP) is also ensured: a telemetry format comprises a number of frames which the system packetises and distributes individual telemetry frames, and also builds telemetry formats for distribution, thus allowing the data to be accessed from either frames or formats as appropriate.
- Behavior checking (limit checking with soft and hard limits, fixed status checking and status consistency checking) with up to 16 behaviors per parameter, selected by applicability criteria. Calibration, with calibration curves of up to 16 points.
- Derivation of synthetic parameters (parsing, interpretation or execution of pre-compiled OL)
- Processing of super-commutated parameters
- Parameter display (32 & 64 parameter alphanumeric display, scrolling, graphics, mimics, Telemetry Query and PUS variable packet displays)
- Out-of-limits display
- Telemetry printouts (full and summary telemetry printouts)
- Telemetry processing control (filing enable/disable, behavior checking enable/disable, reset status consistency, set user-defined constant parameters)
- Recording of OL computation within telemetry packets.

Telecommanding:

- Manual stack
- Any PUS commands, command sequencing including command grouping and blocking
- Pre-transmission command validation (using telemetry status for validation)
- Telecommand encoding
- Command multiplexing for connected command sources (e.g. manual stack, auto stack, external command sources)
- Telecommand uplink
- Post-execution command verification (serveral stages)
- Telecommand history display
- On-board queue model management and display
- Telecommand Query display (manual stack and TC history)
- Telecommand processing control (uplink chain selection, spacecraft register image editing)

Database:

- Online modifications of calibration curves, limits, graphics definitions and telecommand sequences. Editing of the full mission database is not actually a function of the spacecraft control system, but instead the database is imported into the system into an internal format. For the MARECS/MTP system, an Oracle-based editor was developed which generated the required files for import. The Integral MCS will use an MS-Access based solution.

Ground station interface:

- Telemetry, telecommand and station monitoring link configuration, automatic telemetry link reconfiguration
- Recorded telemetry playback from station
- Station operator message send and receive
- Ranging functions.

System level functionality

- User log-in / log-out
- Privilege handling and passing
- Online server selection
- Operator message passing

Events and Actions:

- Events/ alarms routing and display
- Alarm acknowledgement
- Action triggering (e.g. page operational personal, send a command etc.)

Filing and archiving:

- High performance data storage and distribution.
- Administration Utilities available

Flight dynamics interface:

- Live and retrieval packet interface to flight dynamics system (Oratos)
- Interface provided to Flight Dynamic on the basis of executable Telecommand Parameter Files.

Application of Scos 2000 to The Integral mcs

The Integral Mission

ESA's new gamma-ray astronomy mission INTEGRAL (International Gamma-Ray Astrophysics Laboratory) will observe gamma-ray sources located within our Galaxy and across the Universe. Gamma-ray astronomy explores the most energetic phenomena that occur in nature and sources include among others exploding stars, black holes, gamma-ray bursts and pulsars. Not only do gamma-rays allow us to see deeper into these objects but the bulk of the power radiated by them is often at gamma-ray energies of typically 1 Million electron volts. Gamma-rays are produced for example through high-energy particle collisions, nuclear reactions, radioactivity, and interactions between particles and electromagnetic fields (Ref 3).

The instruments on board INTEGRAL will consist of a gamma-ray imager, gamma-ray spectrometer, X-ray monitor and optical camera. These will pinpoint the locations of gamma-ray sources to few arcminutes and measure their radiation energy with an unprecedented accuracy of about 0.2%. This can be achieved today by the use of coded aperture (pinhole) masks and gamma-ray detectors made out of Germanium which are cooled down to - 120 °C and room-temperature solid state Cadmium-Telluride (CdTe) detectors.

The Integral observatory is to be launched in 2001 (Russian Proton launcher). The instruments will be provided by large international scientific collaborations from ESA member states, USA, Russia and other countries. NASA will contribute its deep space network of ground stations to assist in tracking the satellite and gathering the data it records. The nominal lifetime of the mission is 2 years, an extension of up to a maximum of 5 years is technically possible. Most of the observation time will be made available to the scientific community at large. Integral is a complex mission, both in terms of the spacecraft design and the mission design. This therefore requires a reasonably complex ground segment. The Ground Segment is made up of two main components - Operations Ground Segment and the Science Ground Segment as shown in the Fig. 3.

The Integral Science Operations Centre (ISOC) - based at ESTEC (NL) - is responsible for analysing and processing observation proposals received from the science community and incorporating them into the observation plan. It interfaces with the Flight Dynamic System (FDS) in order to generate the plan.

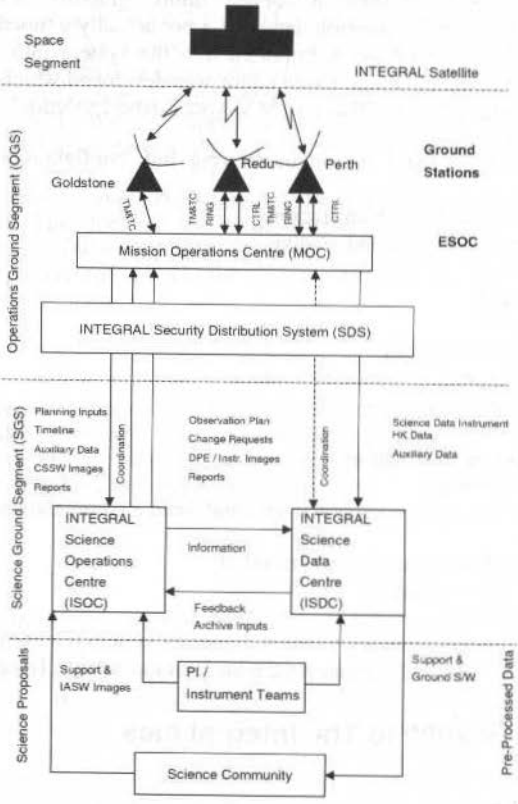


Fig. 3 The Integral Ground Segment

The Integral Science Data Centre (ISDC) - based in Geneva - is responsible for processing science telemetry data, housekeeping data and auxiliary data, in order to produce the data products to be delivered to the science community.

The Integral Security Data System (ISDS) - located at ESOC - this protects the MOC against unauthorised access.

The Integral Mission Operations Centre - located at ESOC - is described in the following sections.

The Integral Mission Operations Centre

The Mission Operations Centre (MOC) at ESOC will be used by Mission Operations Staff to perform spacecraft operations. The MOC is made up of the following components, which interface to the ISOC and the ISDC. The ISDS provides the security access to the MOC.

- **Integral Mission Control System (IMCS)** - used to perform the real time and near real time monitoring and control functions associated with the spacecraft control - this is the part of the system that is based on SCOS 2000
- **Flight Dynamics System (FDS)** - performs the spacecraft orbit and attitude determination calculations
- **On-Board Software Maintenance System (OBSMS)** - the facility where software images are generated and maintained as part of the Integral MOC. It interfaces with the IMCS On Board Software tool (IOBST) by delivering RAM uplink images which are encoded as commands using information from the operational database (IMIB) and uplinked under control of the IMCS commanding subsystem. It also receives dump images from the IOBST
- **Network Controller and TM Router System (NCTRS)** - interfaces with the ground station equipment to provide telemetry, telecommanding, tracking and station computer support to the

IMCS and to receive telecommand requests from the IMCS. It is also responsible for configuring communications links to the different ground stations

- **Flight Operations Plan Generator (FOPGEN)**

The Integral Mission Control System

The Integral Mission Control System derives much of its functionality from the SCOS 2000 system. The IMCS is broken down into the following sub-systems:

- **Databases**, covering importing spacecraft databases from the Spacecraft Prime Contractor, editing and consistency-checking them, and then releasing them to other system components.
- **Telemetry**, covering receiving, extracting and processing housekeeping and PUS service telemetry, time correlation, TM displays and printouts, ranging and tracking, and miscellaneous external interfaces.
- **Telecommanding**, covering requirements related to manual commanding, command validation, transmission and verification, automated schedule execution and display, command history, etc.
- **Mission Planning**, covering requirements related to the generation of command schedules and timeline summaries.
- **On-Board Software Maintenance**, covering requirements related to importing memory images, processing TC images, checksum calculation, OBSM displays and printouts, generation of TM images, and onboard software file configuration control. (Note: The actual on-board software maintenance is performed by the OBSMS, which is not part of the IMCS).
- **Mission Archive and Data Distribution System**: covering the telemetry and auxiliary file distribution system as well as mission archiving on hard medium storage devices
- **Performance Analysis**, covering requirements related to the retrieval, manipulation and display of long-term historical data.
- **Integral File Transfer System,** covering requirements to allow the automated delivery of files throughout various systems of the Integral ground Segment, using TCP/IP and FTP protocols.

Physical Architecture

The Physical Architecture - showing the IMOC, ISDS, ISOC and ISDC is shown in Fig. 4. The main IMCS Server is implemented as a prime and a back up machine. The main IMCS client workstations are the MCR, PSR, DCR and the SSR. The following subsystems have dedicated workstations/servers - the Performance Analysis (PAS), the Archive (MADDS), the Mission Planning (MPS), the Flight Operations Generation (FOPGEN), the ground station interface control (NCTRS), the Flight Dynamics (FDS), the On Board Software SDE from Alenia (the Prime Contractor) is included in the configuration for preparation of on board software updates for uplinking through the IOSBT function of the IMCS). The diagram also shows the remote links to the OSDC and the ISOC as well as the Spacecraft Simulator (on the SIMUSRLAN).

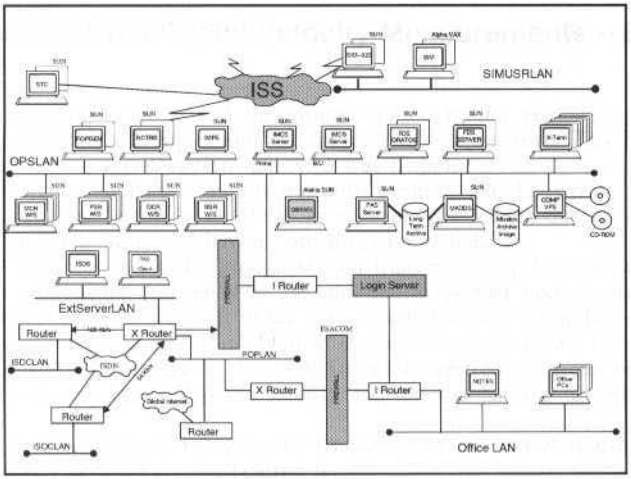


Fig. 4 Integral MCS – Physical Architecture

Conclusion

The selection of the SCOS 2000 system for the Integral Mission Control System represents a significant milestone in the maturity of the system - the Integral case is a highly complex and demanding mission. The SCOS 2000 system is based on a sound open architecture. With a high degree of adherence to industry standards, it will prove to be easy to adapt it to new de-facto standards, both in Information Technology and Space Technology, as they emerge. These possibilities have been discussed in detail in an earlier paper (Ref 4), which covers additional on-going work to evaluate the use of newer technologies (Java etc) in such systems.

References

M. Jones, A. Baldi, C. Lannes, B. Melton, M. Bandecchi, M. Schick, Paper for Spaceops June 1998, "How a Mission Control System was achieved at low cost for a simple mission.", TEAMSAT –Tokyo/Japan

B. Santos, P. Maigné, June 1998, "Use of a generic control system for different types of missions.", Paper for Spaceops, Tokyo/Japan

Integral ESA Science Web Site, http://sci.esa.int/integral/

S. Haag, M. Jones, Spaceops, June 1998, "Use of New Technologies in Flight Control Systems.", Tokyo, Japan.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 113-117

ISSN 0100-7386
Printed in Brazil

Ranging Rate Equipment for the Brazilian S-Band Tracking and Control Ground Stations

Marcus Vinicius Cisotto
Carlos Alberto Ferrari
Instituto Nacional de Pesquisas Espaciais
Av. dos Astronautas, 1758
12227-010- São José dos Campos, SP Brazil
marcao@dss.inpe.br
ferrari@dss.inpe.br

Abstract

This paper presents the architecture and some practical measurements of the Ranging Rate Equipment (RRE), developed at the Instituto de Pesquisas Espaciais (INPE) by the Ground System Division Group, and installed in the Brazilian S-Band Tracking and Control Ground Stations located in Cuiabá and Alcântara. RRE performs the measurement of the Doppler frequency shift in the telemetry receiver second local oscillator output at 180 MHz. These data are used to determine the satellite radial velocity and the orbit parameters, using an algorithm developed by the Flight Dynamics Group of INPE. The equipment can measure frequency Doppler shift (f_D) up to ±150 kHz with a resolution of 6mHz and an accuracy of 30 mHz at a rate of 1 measurement per second. With a rate of 1 measurement each 10 seconds the accuracy is 3 mHz. The measurements are time tagged with a accuracy better than 30 μs using the Universal Time Clock. RRE is based on a standard IBM Personal Computer and can be operated in local mode or in remote mode via a Local Area Network (LAN). The remote control and data transmission use the European Space Agency (ESA) Station Data Interchange Document (SDID) standard messages. RRE also performs the collection of the environmental data (temperature, pressure and humidity) that are sent in the SDID message and are used for correction of tropospheric delays in the process of orbit determination. Measurements and results were obtained using the SCD1 satellite, with the transponder configured in non coherent (one way Doppler) or coherent (two way Doppler) mode.

Keywords: *Doppler Frequency Shift; Radial Velocity, Ranging Rate.*

Introduction

Among the methods used for satellite orbit determination, the measurement of the distance (ranging) from the ground station to the satellite and its rate of variation with time (ranging rate) have been widely employed due to the high accuracy of these methods.

Since the SCD1 satellite launch, INPE has been using only ranging systems for orbit determination with successful results.

To improve the orbit determination process two RREs were developed and installed at INPE's Tracking and Control Ground Stations located in Alcântara and Cuiabá. This paper presents the functional description of RRE and the results obtained.

Orbit Determination from Radial Velocity Measurements – Basic Concepts

The velocity between the satellite and the ground station (radial velocity) can be obtained by measuring the carrier frequency Doppler shift in the communication link. This measurement can be performed in two modes: "one-way Doppler" and "two-way Doppler".

In the "one-way Doppler" mode the satellite transmits a carrier with known frequency whose shift is measured at a ground station. In this mode there are measurement errors due to the satellite on board oscillator uncertainty and instabilities, which are sometimes unpredictable.

In the "two-way" Doppler mode the ground station transmit an up link carrier that is coherently converted to the down link carrier frequency by the satellite transponder and retransmitted to the ground station. This station then measures the shift in the path "ground station – satellite – ground station". In this mode a better measurement accuracy is obtained because a sample of the transmitted frequency is continuously available and also because a high stability reference frequency can be used at the ground station. This method can only be used if the satellite transponder has the capability to operate in a coherent mode.

In the "two-way Doppler" mode, being f_T the frequency of the carrier transmitted by the ground station, then the frequency f_R of the received carrier is:

$$f_R = \frac{p}{q}\left(\frac{c+v}{c-v}\right) f_T$$

where v is the radial velocity, p and q are conversion constants of the coherent transponder and c is the free space speed of light.

The Doppler shift frequency is given by:

$$f_D = f_R - f_{R0} = \frac{p}{q}\left(\frac{c+v}{c-v} - 1\right) f_T$$

where f_{R0} is the received frequency when the relative velocity between the satellite and the ground station is zero.

$$v \cong \frac{cq}{2pf_T} f_D$$

As v<<c the above equation can be simplified, and the radial velocity computed according to the following expression:

$$v \cong \frac{f_D}{f_S} c$$

In the "one-way Doppler" the radial velocity is given by: where f_S is the frequency transmitted by the satellite.

Ranging Rate Functional Description

RRE uses the 'Integrated Doppler' or 'Non-Destructive Range Rate' technique to perform the measurements. With this technique the measurement is performed by measuring the time interval of an integer number of periods of the received carrier allowing measurements with high resolution.

Figure 1 shows the measurements time diagram. The Doppler frequency shift (f_D) measurements are done in an input signal (F_{in}) with a frequency offset of 1 MHz. The Time Measurement Interval (TMI) is a multiple of the 1 pulse per second (PPS) signal received from the ground station time reference system. The measurements instants t(i) are synchronized with the positive edges of TMI pulses. The TMI period (T) can be configured as 1, 10 or 100 s. In the i-th measurement the total number of periods N_i of the input signal is computed. At the i-th measurement instant, windows w(i) are generated. The window starts at the positive edge of the TMI signal and ends at the first positive transition of the input signal (F_{in}) just after the positive transition of the TMI. The i-th window period ($t_w(i)$) is determined using a counter with 179 MHz clock as shown in Figure 2. In this way the input frequency at instant t(i) can be obtained according the following expression:

$$F_{in}(i) = 1MHz + f_D(i) = \frac{N_i - N_{i-1}}{T + t_w(i) - t_w(i-1)}$$

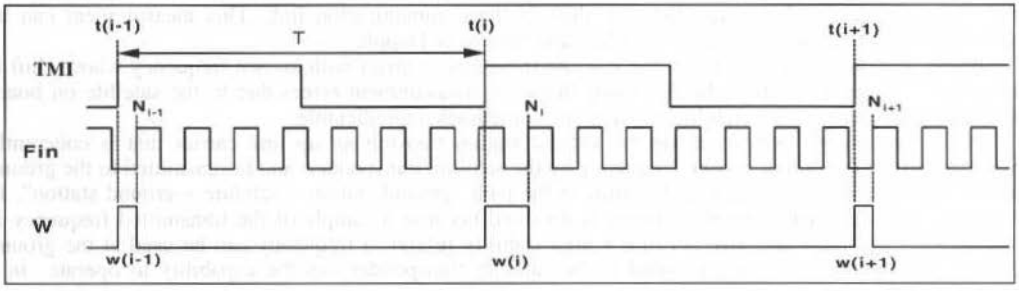


Fig.1 Measurements time diagram

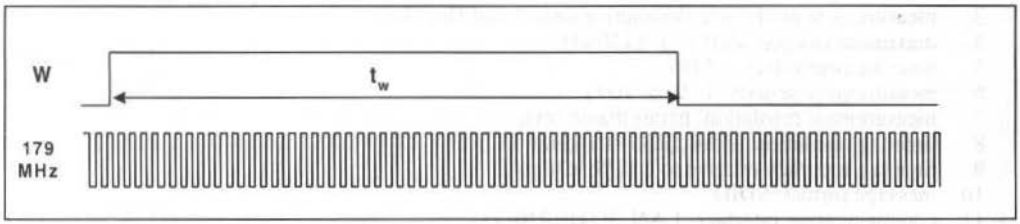


Fig. 2 High frequency clock counter time diagram

At each measurement instant an interrupt request is delivered to the RRE software which reads and stores the counters results and time tag data with an accuracy better than 30μs.

Figure 3 shows a simplified block diagram of RRE. The RRE is based on a standard IBM Personal Computer. The dedicated boards developed for the equipment are plugged into the PC ISA Bus. The RRE software was developed using the Visual C++ for Windows 95 Operational System.

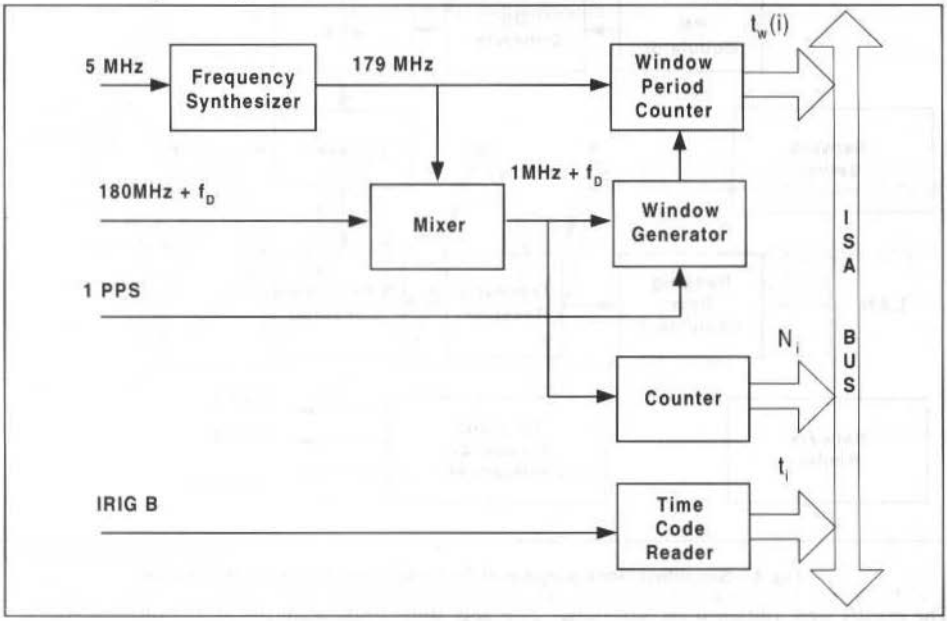


Fig. 3 RRE simplified block diagram

The measurements are performed at the telemetry receiver second local oscillator output at 180 MHz + f_D, which is frequency converted to 1MHz + f_D using a mixer and a frequency synthesizer at 179 MHz. The frequency synthesizer is based on a phase locked loop synchronized with the 5MHz ground station reference frequency. The internal synthesizer also serves as a high frequency clock for the Window Period Counter. With this clock is possible to achieve a measurement resolution of about 6mHz.

The accuracy of the measurements depend on the phase noise of the signal delivered to RRE. This phase noise depends on all the subsystems involved in the communication link. In ground station short loop tests the accuracy obtained was better than 30mHz. In the typical operational conditions of SCD1 and SCD2 the accuracy is better than 180 mHz when the transponder is configured in coherent mode.

The RRE Software other functions are: to format the measurements according SDID standard, to monitor and control the equipment and to transmit the measurements to the Satellite Control Center.

Basic Technical Specifications

The main specifications of the RRE are:

1. input signal: 180 MHz + f_D, 0 dBm, 50 Ohms
2. number of inputs: 2

3. measurement mode: non destructive (integrated Doppler)
4. maximum Doppler shift (f_D): ±150 kHz
5. measurement offset: 1 MHz
6. measurement periods: 1, 10 or 100 s
7. measurement resolution: better than 6 mHz
8. time tag resolution: 1 ms (IRIG-B 1 kHz)
9. time tag accuracy relative to 1 PPS: <30 μs
10. message format: SDID
11. communication interface: LAN (ETHERNET)

Results

The practical results were obtained using the INPE's Tracking and Control Ground Stations located in Cuiabá and Alcântara and the SCD1 satellite.

The Figure 4 shows a simplified block diagram of RRE integrated in the ground station.

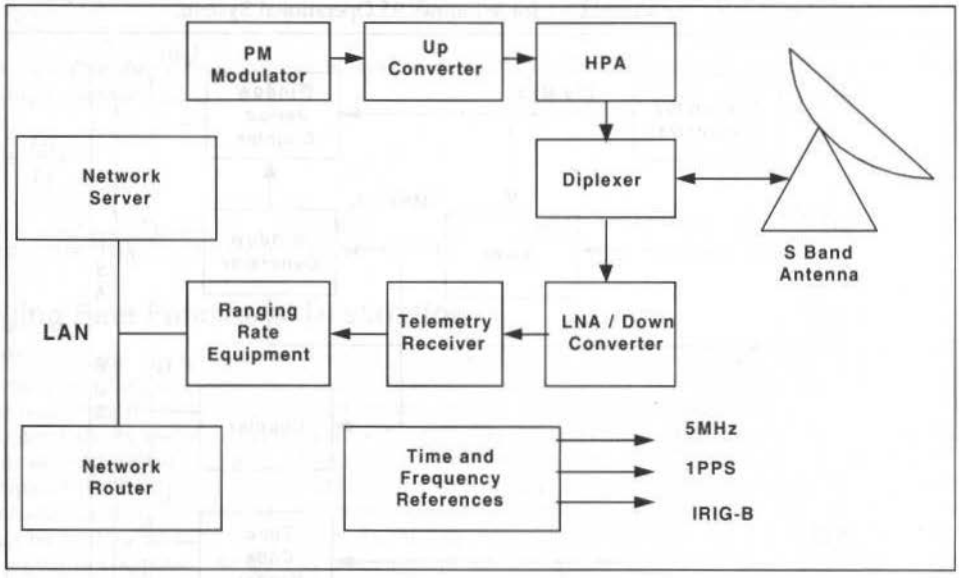


Fig. 4 Simplified block diagram of RRE integrated in the ground station.

The results were obtained on November 29th and 30th, 1998 using the SCD1 satellite configured in coherent mode (two-way Doppler). SCD1 is a spin stabilized satellite (49.02 rpm), with a nominal circular orbit of 750 km and a 25° inclination.

To determine the orbit using only the Doppler measurements an initial estimate of the orbit is necessary. In our case the initial estimate of the orbit was obtained from the ranging measurements. From the determined orbit is possible to calculate the expected Doppler values. In this way we can calculate the residual Doppler, that is the measured Doppler subtracted from the expected Doppler. The residual Doppler is due to random errors (noise, short term frequency instability, etc), bias errors (long term frequency instability, earth station geographic coordinates accuracy, etc) as well to the satellite spin in the case of SCD1 and SCD2.

Compared with orbit determination based on ranging measurements the difference found in the orbit determination using ranging-rate measurements was 232 m RMS in position and 0.29 m/s in velocity. The measurement period used was 10s.

As an example, Figure 5 shows the SCD1 radial velocity calculated from the Doppler measurements data obtained on 30th November, 1998, from 9:48:25h to 9:59:35h.

Figure 6 shows the residual Doppler where is possible to note the effect of the satellite spin on the measurements, as can be seen comparing the residual Doppler with a 49.02 rpm sine curve. From this Figure it is also possible to verify that the equipment accuracy is better than 3 cm/s.

The results obtained with the SCD2 are similar.

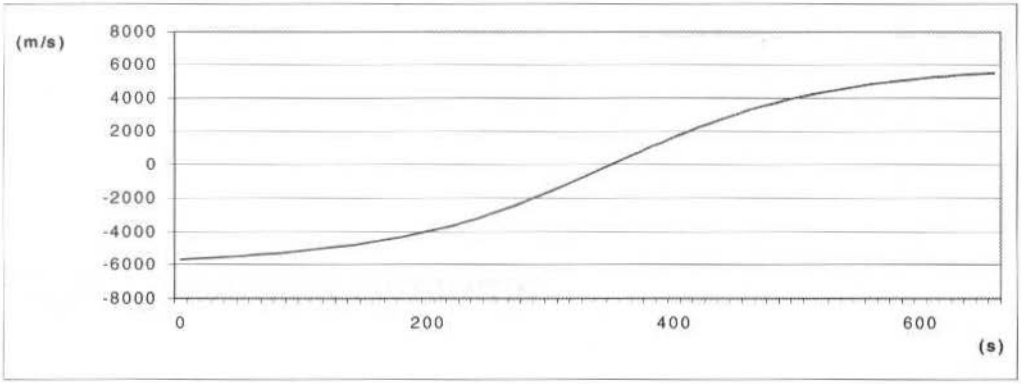


Fig. 5 SCD1 radial velocity

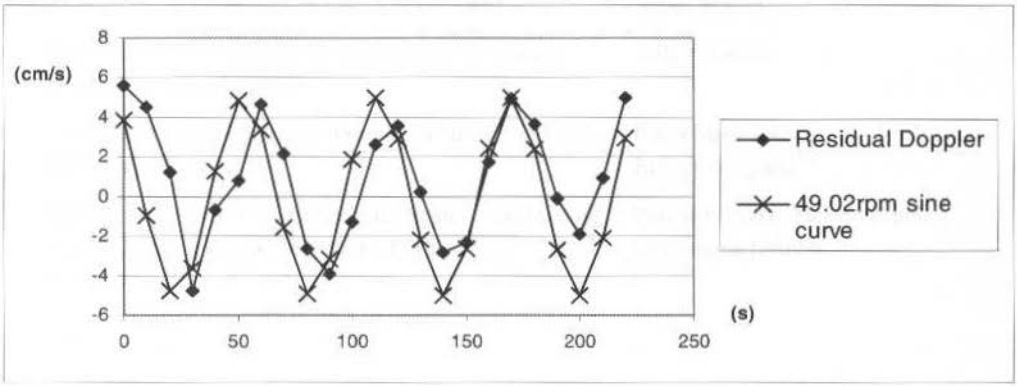


Fig. 6 Residual Doppler

Acknowledgments

We would like to express our sincere gratitude to Dr. Helio Koiti Kuga, of the INPE's Fligth Dynamics Group, for his support and the analysis of the results.

Conclusions

This paper has presented the functional description and general characteristics of RRE developed for INPE's S-Band Tracking and Control Ground Stations. The results obtained using the SCD1 satellite show that the equipment performance is within its specifications.

References

Fukuda, F. H., 1986, "Desenvolvimento de Um Equipamento para Medida da Velocidade Radial de Satélites", Master Dissertation, INPE.

Gaudenzi, R., Lijphart, E.E. and Vassallo, E, 1990, "The New ESA Multi-Purpose Tracking System", ESA Journal, Vol. 14.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 118

ISSN 0100-7386
Printed in Brazil

Flight Dynamics Systems

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 119-126

ISSN 0100-7386
Printed in Brazil

Launcher Operations and Satellite Navigation: Safety-Critical Flight Dynamics Software in the 1990'S

Philip Davies
Stuart Martin
Logica Space Division
Wyndham Court
74 Portsmouth Road, Cobham
Surrey, KT11 3LG, England
daviespe@logica.com
martinsd@logica.com

Abstract

As the Space industry moves into new areas of activity such as the provision of navigation data for civil aviation, the amount of safety critical software developed within the industry is ever-increasing. In addition to the new application areas, existing safety-related systems are being replaced. In all of this work the usual commercial constraints of schedule and budget must be observed whilst maintaining system safety.

Logica have a wide experience of working on safety-related and safety-critical systems in a variety of industries. Within the Space industry Logica have experience in several programmes containing safety-critical flight dynamics software. This paper looks at two of these programmes:

The first of the projects is the Système de Coordination de Traitement et de Visualisation, (Goodwin, A, et al.: 1997) (SCTV) which is part of the CNES CSG2000 programme to modernise the launch facilities for Ariane 4 and 5, providing the ground infrastructure for the Ariane launcher program for the next fifteen to twenty years. The main purpose of the SCTV system is to monitor the trajectory of Ariane launchers throughout their flight. Radars positioned around the launch site and telemetry from the on-board inertial navigation system feed SCTV with data describing the launcher's position. This information is processed in real-time and the estimated trajectory displayed to the flight safety officers. If the launcher is shown to be deviating in a way that could be dangerous, then the range safety officers have the power to neutralise the launcher in flight before it can cause loss of life or injury to people and equipment on ground.

The second of these projects is the European Global Navigation Overlay System (EGNOS), the European element of the first generation Global Navigation Satellite System (GNSS-1). The purpose of the GNSS-1 system is to augment the existing global navigation and positioning service provided by the American GPS and Russian Glonass satellites. EGNOS will be the European contribution to GNSS-1, working alongside equivalent American and Japanese systems. The enhancements provided by EGNOS are the provision of:

- *a GPS-like geostationary ranging augmentation which will provide users with additional pseudo-range measurements,*
- *an integrity monitoring augmentation which will improve the integrity of the navigation service to users,*
- *a wide area differential augmentation which will improve the position accuracy by broadcasting differential corrections to users.*

The project is safety-related because of the way the service will be employed by users. The most significant user community, from a safety point of view, will be commercial airlines who will use the EGNOS system as sole means for en-route navigation up to Category 1 precision approach.

The paper describes these two projects and compares the approach to the development and testing of the safety-critical flight dynamics software. In particular we address:

- *the impact of safety engineering on the design,*
- *the application of standards in the development,*
- *considerations coming from system certification,*
- *the testing of safety-critical flight dynamics software,*
- *the operational impacts of safety-critical flight dynamics software.*

The projects are at very different stages of development with SCTV operational and EGNOS at the start of its implementation phase. It is interesting to compare the approach we took on SCTV with the approach we plan to take on EGNOS.

***Keywords**: Safety, Critical, Launcher, Navigation.*

Safety Critical Flight Dynamics Software

The following text summarises the two projects and the place of the flight dynamics software within the systems.

SCTV

The Système de Coordination de Traitement et de Visualisation (SCTV) project is part of the CNES CSG2000 programme to modernise the launch facilities for Ariane 4 and 5. It is now operational at the Centre Spatiale Guyanais (CSG) in Kourou, French Guiana, and will continue to be used for the next fifteen to twenty years.

The main purpose of the SCTV system is to monitor the trajectory of Ariane launchers throughout their flight. Radars positioned around the launch site and telemetry from the on-board inertial navigation system feed SCTV with data describing the launcher's position. The flight dynamics software is contained within the computation centre (CCEL) and processes this information in real-time during the flight. The estimated trajectory is displayed to the flight safety officers. If the launcher is shown to be deviating in a dangerous way, then the range safety officers have the power to neutralise the launcher in flight before it can cause loss of life or injury to people and equipment on ground.

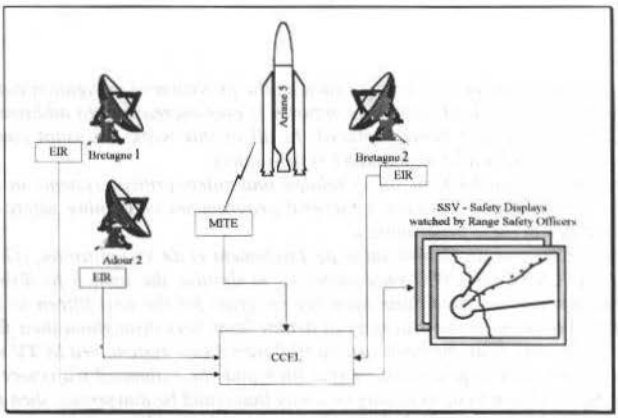


Fig. 1 Overview of the SCTV system

SCTV is distributed across several sites and over several computers within each site. The principal components of SCTV's real-time system are:

- the local CSG radar interface equipment (EIR), one per radar site (Adour2, Bretagne1, Bretagne2),
- the redundant telemetry centre interface equipment (MITE),
- the localisation processing and co-ordination sub-system (CCEL), which includes the primary and secondary trajectography processors (TR1, TR2), the external interface equipment (IS) and the radar and system monitoring and control displays (RCR, RCTDL). The majority of the flight dynamics software runs on the dedicated trajectography processors TR1 and TR2,
- the real-time flight safety sub-system (SSV), including the primary and secondary displays (RSV, ISV).

SCTV was implemented between 1994 and 1996 and, after a year of qualification, has been operational since late 1997.

EGNOS

The purpose of EGNOS is to enhance the existing global navigation and positioning service provided by the American GPS and Russian Glonass satellites. The European Geostationary Navigation Overlay System (EGNOS) will be the European contribution to GNSS-1, working alongside equivalent American and Japanese systems. The enhancements provided by EGNOS are the provision of :

1. a GPS-like geostationary ranging augmentation which will provide users with additional pseudo-range measurements,
2. an integrity monitoring augmentation which will improve the integrity of the navigation service to users,
3. a wide area differential augmentation which will improve the position accuracy by broadcasting differential corrections to users.

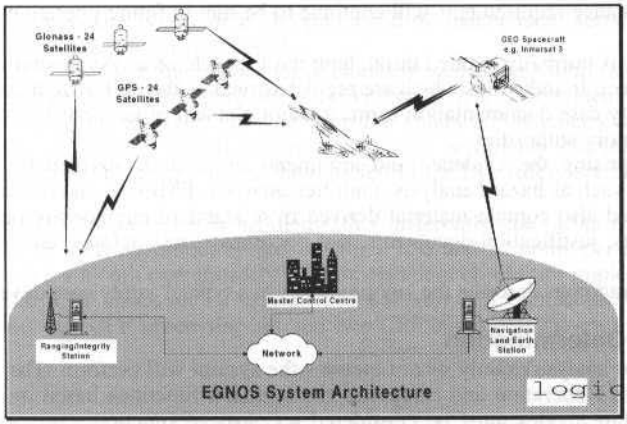


Fig. 2 EGNOS System Architecture

As a navigation system, EGNOS works by receiving GPS, GLONASS and GEO signals at a network of ground ranging and integrity monitoring stations (RIMS). The raw data received by the RIMS together with the RIMS own observables of the satellites in view are passed over the wide area network into a set of redundant master control centres (MCC).

Each MCC contains a central processing facility (CPF) and a central control facility (CCF). The CCF provides the operators tools for monitoring and controlling EGNOS. The CPF is the algorithmic heart of the system and contains the majority of EGNOS' real-time flight dynamics software. It derives all the information necessary to support the generation of the EGNOS broadcast bit stream.

After its computation in the CPFs, the resultant EGNOS Signal In Space is uplinked to the geostationary satellites e.g. INMARSAT 3 AOR-E and IOR and by a set of navigation land-earth stations (NLES).

Thus, the message computed in the CPF and transmitted to the user from the GEO via the NLES contains the following information:

- Various correction parameters that enable the user segment to counter errors in the satellite clock and ephemeris broadcast in the GPS/GLONASS navigation messages.
- Parameters that enable a more accurate calculation of the ionospheric delay than is currently possible, thus enabling single frequency (L1) users to realise their own position with greater accuracy,
- GEO ephemeris parameters - orbit and clock - that enable the user segment to make use of the single frequency (L1) GPS look-alike ranging signal transmitted by the GEOs in their position computations,
- Parameters relating EGNOS Network Time (ENT), GLONASS time and UTC,
- Parameters describing the accuracy of the various corrections.

EGNOS implementation phase started in late 1998 and the system is planned to be operational 5 years later in 2003.

The Impact of Safety Engineering on the Design

System Safety Case

Safety critical software always forms part of a system which encompasses other software and hardware. The software is called "safety critical" because it has the capability of putting the system into a hazardous state.

Safety engineering is principally concerned with making sure the system being designed, built and operated is safe to use in its intended operational scenario. The means by which safety is proven is usually called the *safety case*.

There are different approaches to the safety case in different countries and industries but in all cases the safety case should be the mechanism used to present the evidence, arguments, assumptions and methods used to show that safety has been properly addressed during design, development and operations of the system. It must show that the system's performance requirements have been met. Further, it must

show that the performance requirements will continue to be met in future operations and in non-nominal conditions.

The methodology is normally applied throughout the lifecycle of a system starting from a safety plan very early in the project. In industries which are regulated such as the air traffic management industry the reviewing of the safety case documentation forms a major element of the certification process performed by the relevant regulatory authorities.

The details comprising the evidence and arguments are usually derived by standard safety and reliability techniques such as hazard analysis, fault tree analysis, FMECA, and reliability block diagrams. The safety case should also contain material derived from standard engineering practice - specification and design documents, justification documents, test documents, test outputs, calculations and whatever else is needed.

The following paragraphs describe the key processes in a typical safety case development.

Safety Function Determination

The first step is to analyse exactly what functions the system will perform. The aim of this step is to understand the system's operation and categorise the system's functions based on the consequences of failure. Different failure modes must be considered e.g. loss of function, corruption of function. The output of this stage is a "severity category" for each of the functions.

Hazard Analysis

The next step is the hazard analysis. From the functional analysis in the previous step we identify the system hazards. A mapping between the system hazards and the functions then allows an overall category to be allocated to each of the system hazards.

Causal Analysis

Having identified and categorised the system hazards the next step is to perform the probabilistic analysis which leads to the computation of the likelihood of the hazard occurring. The most common method for doing this "causal analysis" is fault tree analysis. The fault tree is a hierarchical decomposition of the hazard into lower level events. The tree is taken far enough down until the it is possible to allocate reliability data to the base events. This is usually to the level of replaceable units for which mean time between failure (MTBF) and mean time to repair (MTTR) data are available. It is then possible to compute the probability of the top event using the data in the base events and the logical connections between the events (AND gates, OR gates).

Risk Tolerability Analysis

For each system hazard we know its categorisation from the hazard analysis and its frequency of occurrence from the causal analysis. These two attributes allow us to produce a "Risk Matrix". The risk matrix is a graphical representation of the frequency of occurrence versus the severity of the hazard. A typical risk matrix template is shown below:

	Cat 1	Cat 2	Cat 3	Cat 4
Frequent	A	A	A	B
Probable	A	A	B	C
Occasional	A	A	C	D
Remote	A	B	D	D
Improbable	B	C	D	D
Incredible	C	D	D	D

The space is divided into regions each of which is assigned a tolerability:

- A, being unacceptable,
- B, being undesirable and acceptable only in exceptional circumstances,
- C, being acceptable but only with the endorsement of the operating authority,
- D, being acceptable.

The aim of this analysis is to demonstrate that all of the hazards are acceptable i.e. are in region D. For those hazards in region C we must show that the risk is "as low as reasonably practicable" (ALARP) i.e. we have taken all reasonable measures to reduce the risk and to reduce the risk any further the cost would be out of proportion with the benefits gained. Regions A and B are generally unacceptable.

The safety analysis of both projects indicated that the most critical system hazard is the output of hazardously misleading information i.e. a loss of integrity. The most probable cause of this in both projects was a failure of the flight dynamics algorithms and this is where the greatest impact on the design was felt. Essentially, to achieve a high resilience against non-integrity it was necessary in both

cases to ensure independent data is available and processed by independent software in order that the results being output can be independently checked.

In the case of the SCTV project the independence comes from different sources of trajectory data. In addition to the three sets of radar data, there are also two completely independent streams of telemetry data. During the critical phase of flight, the data from the radars and telemetry are continually cross-checked via the MMI, so that any clearly incorrect data can be quickly eliminated from the chain. Integrity is further maintained by ensuring that the data fed to the two safety screens is also completely independent.

In the case of the EGNOS project it is handled rather differently due to the nature of the system - EGNOS is designed to be used operationally 365 days per year and 24 hours per day while SCTV is a system used once or twice a month operationally. The independent checks are implemented by a separate software suite called the "CPF check set". The main CPF software is known as the "CPF processing set" and each MCC contains one CPF processing set and two CPF check sets driven by independent RIMS data.

Partitioning

The safety engineering work performed in the requirements analysis and design phase produces, amongst its outputs, criticality assessments of the functions and components making up the system.

The Application of Standards in the Development

Introduction

During the 1980's the importance of software standards was recognised in many safety critical industries including the aerospace industry. The work led to the development of the key software standard for software which runs on-board aircraft, DO-178/ED12. The relevance of this standard to ground software is discussed below. We also discuss the recently ratified international standard for safety critical software relevant to *any* industry, IEC 61508.

RTCA/DO-178B / EUROCAE ED-12B : Software considerations In Airborne Systems And Equipment Certification

This document is a joint recommendation from EUROCAE and RTCA that has been adopted and made mandatory by many states. The production of the document is the result of collaboration between Radio Technical Commission for Aeronautics (RTCA) and the European Organisation for Civil Aviation Equipment (EUROCAE); formerly, the European Organisation for Civil Aviation Electronics. RTCA is an association of interested parties drawn from the government and industry of USA. EUROCAE is based on a similar source of contribution but from Europe.

Ostensibly covering software of airborne systems, it does not relate directly to ground or space segments. However, it is influential in those areas and could be construed as forming part of "best practice". It includes checklists of methods to apply or avoid. In particular it places requirements on:

- the software planning process,
- the software development processes,
- verification,
- testing,
- configuration management,
- quality assurance,
- certification liaison.

The purpose of the document is to provide guidelines for the production of (airborne) equipment so that the equipment performs its intended function with a level of confidence that complies with airworthiness requirements. The document provides a process-focused set of guidelines and addresses the certification of equipment used on aircraft where the equipment incorporates software. It has become a de-facto standard for the development of the equipment in question.

The document identifies five categories of failure of airborne equipment; each category determined by the impact of the occurrence of failure. A safety assessment process is identified as a means to determine the contribution of software to potential failure conditions and this determines the Level of Software required for the system. The Level of Software determines the effort required to show compliance of software with certification requirements which are, in turn, determined by the category of any failure. The guidelines cover:

- objectives for software lifecycle processes,
- descriptions of activities and design considerations for achieving those objectives,
- description of evidence that indicate that the objectives have been satisfied.

The document is well known to certification authorities, and is generally well regarded.

International Standard IEC 61508 Parts 1 to 7 - Functional Safety : Safety Related Systems.

This ISO Standard is intended to be applied to the development of safety related systems when part or all of the system includes programmable electronic devices. It is not specific to any application such aerospace.

It is an extensive document, covering a wide range of safety management activities in detail. It sets a lower limit on the target failure measures, in a dangerous mode of failure, that can be claimed for a single E/E/PES safety-related system. For safety-related systems operating in:

- a demand mode of operation, the lower limit is set at 10^{-5} (probability of failure to perform its design function on demand),
- a continuous mode of operation, the lower limit is set at 10^{-5} (probability of a dangerous failure per year).

The standard uses a Safety Lifecycle Model for all activities necessary for ensuring that the required Safety Integrity Levels (SIL) are met for the Safety Related System. Under the standard, safety related systems are categorised as one of four SIL depending upon the requirement for Probability of Failure to Perform either on Demand or under conditions of Continuous Operation. These are shown below.

Table 1 Safety Integrity Levels: Target Failure Measures

SIL	Demand Mode of Operation (Probability of failure to perform its design function on demand)	Continuous/High Mode of Operation (Probability of a dangerous failure per year)
4	$>=10^{-5}$ to $<10^{-4}$	$>=10^{-5}$ to $<10^{-4}$
3	$>=10^{-4}$ to $<10^{-3}$	$>=10^{-4}$ to $<10^{-3}$
2	$>=10^{-3}$ to $<10^{-2}$	$>=10^{-3}$ to $<10^{-2}$
1	$>=10^{-2}$ to $<10^{-1}$	$>=10^{-2}$ to $<10^{-1}$

As a result of a process of Hazard Analysis, an overall Safety Requirements Specification for the system is to be developed covering both Safety Functions Requirements and Safety Integrity Requirements. the reduction of external risk is also addressed. In addition to the Lifecycle, the standard addresses safety management and safety assessment. In particular, a safety Plan is to be developed as that part of the Quality Plan that focuses on safety. As a further part of the safety management process, the competence required of persons involved in safety activities within the safety lifecycle are addressed.

A key feature of the standard is that techniques and measures that might be used in the development and implementation of a safety related system are ranked in terms of the likelihood of their leading to a realisation of the required level of integrity of the system. In particular, software development techniques are assessed for their suitability for a system of a given level of integrity requirement.

Considerations coming from System Certification

Certification Liaison

The formal approval that a safety-related system can be used for its intended purpose is known as certification. Certification is the formal declaration that a system is indeed safe to use for that application and complies with its requirements. The certification procedure may take many forms involving diverse activities such as audits, tests, and trials. It is usually performed by a combination of the organisations

involved in the development, procurement and operation of the system and an independent "certifier" such as a regulatory authority or a part of the organisation independent of the project such as a QA/Safety organisation.

In order that the system can be easily certified it is highly recommended that the system developers liaise with the eventual certifiers from a very early stage in the project and at regular intervals throughout development. The developers should inform the certifiers of their approach to the project and to certification. It is much better that any changes brought about by the certifiers are discovered as early as possible.

Choice of Programming Language

Ada remains the language of choice for safety critical systems. Indeed the IEC 61508 standard highlights Ada as one of its recommended languages whereas "C" is not recommended for safety-critical systems. The SCTV project took the decision that as far as possible all new code would, therefore, be written in Ada including all flight dynamics algorithm implementation. This decision was guided by practical experience on previous Logica projects. It is also our belief that Ada compilers and test tools provide better testing at the early stages of a project, and so reduce integration problems. Stringent project coding standards restricted Ada usage to a "safe subset".

The most safety critical flight dynamics software in EGNOS is the CPF check set software which is classified as DO-178B/ED12B level "B" which is known as "Hazardous/Severe Major". As with SCTV this software will be developed in Ada.

A difference between the two projects is in the choice of which version of Ada. Due to the different starting times of the two projects (mid 1990's vs. Late 1990's) the earlier project used Ada 83 but it is hoped to implement the EGNOS check set software in Ada 95.

The Testing and Operational Impacts of Safety-Critical Flight Dynamics Software

Structural Testing

For the safety critical flight dynamics software in both projects, full code coverage and branch coverage of all source statements was required. For each procedure, sufficient test cases must be scripted to demonstrate 100% statement and condition coverage, with special attention paid to the testing of boundary conditions.

To assist with this task during the SCTV project, the use of the AdaTest product was mandated for testing at both unit and module levels of all critical software. AdaTest works by instrumenting the original Ada source code which, when executed, generates a report containing statistics for all instrumented units. As well as providing information on code and branch coverage for each unit, the report also includes other useful information such as the number of executions of each line, and the overall complexity of each unit according to the McCabe metric.

Employing this mechanism during the early testing provided a number of advantages, principally

- All testing of critical software was done in a consistent manner, making it easy to collate and compare results.
- Tests were easily repeatable following changes later in the project (regression testing), and results easily compared.
- Regression testing could, to a large extent, be automated, hence reducing the associated overhead.
- Identical statistics available for every unit and module, helping the Quality Engineer track adherence to project standards and identify possible problem areas.

Numerical Verification

The numerical verification of the integrated flight dynamics software is achieved in two stages - off-line verification followed by real-time verification.

The aim of the off-line verification is to verify the mathematical correctness and numerical performance of the algorithms.

For example the SCTV project performed comparisons of:

- CSG radar and inertial navigation results obtained off-line with independently obtained study results for selected historical flights and for simulated deviated flights,
- CSG radar and telemetry results obtained in real-time with the corresponding "off-line verified" results.

Similarly, in the EGNOS project we plan to do the verification using both real recorded GPS and Glonass data and also using synthetically generated data which can more easily look at "worst cases".

Real-time Verification

Once the algorithms are proven in the off-line verification it is necessary to prove them as part of a real-time system. SCTV achieved this by injecting the same raw data, either historical or synthetic, used in the off-line verification into the real-time system. For EGNOS the same approach is planned.

The Operational Impacts of Safety-Critical Flight Dynamics Software

The correct functioning of the safety critical flight dynamics software is not always easy to observe. A well constructed MMI should allow the operator to have an insight into the performance of the algorithms. On SCTV the following were implemented to aid the operators:

- display of numerical indicators of estimated accuracy,
- graphical comparisons of independent data,
- display of algorithm internal status.
- Alarms on critical status changes, and when certain parameters reach threshold values

The combination of these sets of information means that the two CCEL operators can very quickly identify problems with any of the incoming data streams, and so prevent them from being used.

Conclusions

The key points are:

- the use of flight dynamics software in safety critical systems is increasing,
- the approach to the development of this software is becoming more formalised as time goes by,
- safety engineering has large impacts throughout the whole project lifecycle,
- flight dynamics software presents particular problems in safety critical systems which come from the algorithmic nature of the implementation.

Acknowledgements

SCTV - we wish to thank B Dellery, B Lacosta and C Richardson of CNES for their help and encouragement.

EGNOS - we wish to thank W Goxe of Alcatel for his help and encouragement.

References

Goodwin, A, Davies, P;, and Harborne, J., June, 1997, "SCTV: Ground Based Trajectography System for Tracking Ariane Launchers", 12th International Symposium on Flight Dynamics, Darmstadt.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 127-132
ISSN 0100-7386
Printed in Brazil

Quartz ++ : Matra Marconi Space new Generation Flight Dynamics System

Didier Breton
Space Mechanics and GNC Department
Matra Marconi Space
31 avenue des Cosmonautes 31402Toulouse cedex. France
didier.breton@tls.mms.fr

Abstract

For the past ten years, MATRA MARCONI SPACE has been accumulating a large experience in the development and operation of Flight Dynamics Systems for geosynchronous satellites station keeping and transfer.
QUARTZ software was used by MATRA MARCONI SPACE for the Launch and Early Operation Phase (LEOP) for ORION F1, NILESAT and ST-1 (ASTRA 2B operations scheduled 1999). COSMIC, a station keeping dedicated Flight Dynamics System is used for NILESAT, ST1 and WORLDSTAR satellites On Station operations.
Based on the experience of these two latter products, MMS has developed a new generation Flight Dynamics System: QUARTZ ++. This new package is designed for both station keeping operations, transfer preparation (Mission analysis) and LEOP. QUARTZ ++ was selected by INTELSAT as part of its new Flight Dynamics and Commanding System, for the control of up to 30 satellites and INTELSAT's future satellites' LEOP operations.
The design of this new Flight Dynamics System is driven by two main goals. On one hand the user-friendliness of such a system shall be maximized in order to diminish the risk of errors and the operators workload. For that purpose, the Flight Dynamics algorithms shall use advanced Graphical User Interfaces (GUI) and database systems. On the other hand, the system design shall be flexible in order to allow Flight Dynamics experts to easily install new flight dynamics applications in the system.
In order to fulfill these two objectives, QUARTZ ++ architecture first consists in a UNIX framework which provides the flight dynamics algorithms with generic functions:

- *an evolutionary database system generic GUI definition for algorithms input and output as well as database edition*
- *data acquisition services (for telemetry as well as track and rage measurement processing)*
- *plot services (based on PV-WAVE).*
- *Reports and broadcast services (transmission of various products through the network)*
- *Flight dynamics algorithms execution management. Both interactive and batch modes are available (batch mode is used for automatic scheduling of various Flight Dynamics tasks).*

Thanks to the framework generic features, the integration of new satellites or new flight dynamics functions is completely handled by the flight dynamics experts.
The flight dynamics functions available in QUARTZ ++ include a complete set of flight proven station keeping and collocation functions including orbit determination, maneuver computation and implementation orbital and sensor event predictions. Flight proven LEOP functions are also available such as transfer planning optimization, attitude determination and attitude maneuvers planning for spin stabilized spacecraft, as well as pre-launch mission analysis tasks such as launch window computation and dispersion analyses.
Thanks to a connection with TIMELINE EXECUTER S/W (operation scheduling software developed by MMS) the next evolution of QUARTZ ++ will feature advanced functions for automatic scheduling of station keeping maneuver for a complete fleet of satellites.
This paper describes QUARTZ ++ Flight Dynamics System, the flight dynamics functions as well as the framework generic services.
Keywords: *Flight Dynamics System, Station-keeping, collocation, transfer, operations.*

Main design goals for a new Flight Dynamics System

The main requirements that drove QUARTZ ++ Flight Dynamics System design result from a 10 year MMS experience in Flight Dynamics System development and operation. The main design goals are summarized hereafter:

- The FDS system shall support both geosynchronous S/C station keeping operation (including collocation) and transfer operation (including pre-launch analyses).
- The system shall cope with the operation of a whole fleet composed of a large number of satellites. Spacecraft in the fleet may have been procured from different manufacturers and may each require specific databases parameters as well as specific algorithms for control.
- The system concept shall minimize the operation workload and minimize the risk of human error. For that purpose, operating the FDS shall rely on user-friendly Graphical User Interfaces (GUI) (conforming MOTIF style guide of point-and-click interface). GUIs look and feel shall be

consistent through the whole system, providing common dialog boxes for similar functions and making the use of the system as much intuitive as possible. In order to minimize the risk of error, manual data typing shall be minimized. Default options shall be provided so that routine operations require very few manual input. Reasonableness checks shall be provided on any manually input value. The organization of GUI and the chaining of the flight dynamics function shall reflect the operational scenario.

- An open, configurable and maintainable product: The system shall allow an easy integration of new flight dynamics functions as well as the addition of a new spacecraft in an existing fleet. The GUI layout, default values and database structures shall be easily customizable. The openness of the system is a key feature in order to install in the new system flight proven flight dynamics algorithms (These algorithms may either be inherited from in-house formerly developed flight dynamics package QUARTZ or COSMIC, or can be customer provided specific algorithms).
- Generic flight dynamics algorithms: As much as possible, the FDS shall be based on generic algorithms (i.e. independent from any spacecraft technological specifics). Since some spacecraft specifics algorithms are mandatory (such as propulsion models), such specific algorithms shall be restricted to a limited number of Software components. This design constraint aims at making easier the adaptatation of the FDS to any new kind of spacecraft: only these custom components are affected.

QUARTZ ++ concept baseline: separation between FRAMEWORK and FLIGHT DYNAMICS ALGORITHMS.

In order to meet the above mentioned objectives, QUARTZ ++ architecture is characterized by a clear separation between the flight dynamics science part and computer science part. As a matter of fact, the FDS is composed of two main components: the *framework* and the *flight dynamics algorithms*.

The framework handles the database system, the GUI generation functions, plot functions, report archives and broadcast, telemetry and track and range acquisition aspects. The framework manages the execution of The flight dynamics algorithms which are generally FORTRAN or C programs. Flight Dynamics Algorithms interface with the framework through simple Application Program Interface (API) in order to get input parameters or output results . A couple of simple syntax ASCII configuration files shall be completed when installing a new algorithm or a new satellite in the system. These configuration files will be interpreted by the framework in order to provide the algorithm with the relevant database parameters, to construct the GUI displays or to provide the flight dynamics function with any other framework generic function like plotting or data acquisition. In a similar way, configuration files are use to depict the database structure for each satellite. This concept allows the flight dynamics experts to handle completely the integration of algorithms or new spacecraft in the FDS.

QUARTZ ++ FRAMEWORK main features:

The main features of QUARTZ ++ framework concept are briefly depicted hereafter. This description is not exhaustive, it only aims at giving the main characteristics of the FDS system.

Database:

For each satellite defined in QUARTZ ++ 6 databases are available: the orbit database, the attitude database, the spacecraft constants database, the spacecraft physical database, the maneuver set database, the ground station set database. Each of these databases include an indefinite numbers of records. A database record is usually called a vector. Typically, the orbit vector includes orbital elements with the corresponding date, drag coefficient, solar radiation pressure coefficient and the associated covariance matrix. The attitude vector includes typically right ascension and declination and associated covariance matrix. The physical vector includes data such as mass, pressure temperature per propellant tank. The spacecraft constant vector is dedicated to technological data. The orbital or attitude maneuvers by various parameters such as start time, duration, thrust profile. In fact, flight dynamics functions do not handle single maneuvers but maneuver sets which include an indefinite number of maneuvers. The basic idea is to include in a single entity all the scheduled maneuvers for a given satellite so that any flight dynamics algorithm which performs orbit propagation or event prediction can take the effect of any scheduled maneuvers into account. In a similar way, the Ground Station set vector includes an indefinite set of ground stations antennas described by parameters such as coordinates or measurement biases. User-friendly GUI functions are available to construct sets of maneuvers or ground station from the maneuvers or antennas available in the database.

For any kind of vector (orbit, attitude, maneuvers...), the definition and the number of the parameters for each vector is configurable and may differ from one spacecraft family to an other. (In particular, the format of the spacecraft constants vector is generally different between two satellite series).

The databases are defined on a satellite basis.

Activities and modules:

Modules are the unitary level of task which can be run separately by the Flight Dynamics user. Several modules can be chained in an activity so that the output of one module can be used directly as input for the next module without transiting through the database. (For example the inclination control maneuver planning activity is split in several modules: among them, a module will compute theoretical delta-velocity correction from the physics, a second module will convert the theoretical DV into start time and duration from the propulsion model) When running an activity, the user may chose to simply run all the modules in a raw, but he can also run them one by one, checking the results of a first module before activating the next one or re-running a particular module several time with different processing options before activating the next one. The task sharing between modules provides the user with a lot of flexibility when operating the FDS.

Furthermore, QUARTZ ++ framework configuration file system allows to easily build new activities by combining modules. For example all station keeping maneuver planning activities have similar module architecture: only the theoretical maneuver computation module will distinguish the inclination control maneuver planning activity from the longitude eccentricity control maneuver planning. In parallel two inclination maneuver planning activity for two different spacecraft may only differ by their implementation module (computation of start time and duration from the propulsion model).

Several users may work in parallel on the FDS, each user performing activities on several satellite or cluster of satellites. (Typically for when operating collocated satellites an activity like the relative motion prediction activity access database data from several spacecraft) . Every user may initiate an indefinite number of activities in parallel.

GUIs:

QUARTZ ++ framework provides generic functions for input and output GUI generation (based on ILOG-VIEWS product). For a particular module, any input parameters may come from a selected database vector, it may also be entered in the module input GUI, it is also possible to have the database value as default value in the GUI and to overwrite it. The general philosophy is to display in the activity input GUI only the parameters that may have to be modified by the analyst such as processing options. (Typically spacecraft constants such as thruster orientation or propulsion model coefficients do not have to be displayed during routine operation. Such parameters are edited separately through the database vector edition GUIs).

QUARTZ ++ GUI generation facility enables GUI layout, default values and bounds for reasonableness checks to be configured and modified easily, for flight dynamics modules as well as for database edition.

Data Acquisition:

Flight Dynamics applications such as orbit and attitude determination process Track and Range or telemetry measurement data. QUARTZ ++ framework provides these applications with a simple interface with Track and Range or telemetry. The framework manages the acquisition and archival of measurement data, acquisition can cope with both deferred time and real time (useful for near real time orbit/attitude determination process based on Kalman filtering).

Plots:

Plot functions rely on PV-VAWE product. PV-VAWE functions are directly invoked from the framework and allows any tabular result to be immediately plotted. Custom plot templates can be defined for the most regularly used plots.

Reports:

Flight Dynamics Activities may produce ASCII report files which may be used for interfacing with other components of aground segment (typically maneuver messages). QUARTZ ++ framework provides functions to view and broadcast these reports through the network to any e-mail or ftp address.

Archive management:

QUARTZ ++ includes means to archive all input and output data as well as all reports generated by an activity. Archives can be easily reinstalled so that the user can re-run that activity in the same input context it was run before. Archives are the primary means to ensure the tracability of operation.

Batch mode:

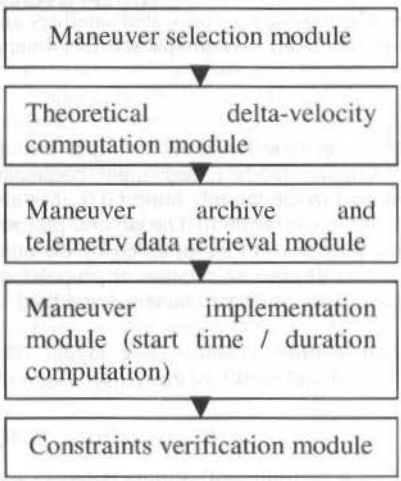
Any activity that can be run in interactive mode (through GUIs) can also be run in batch mode. This batch mode is used typically for routine tasks; for example the batch modes enables to generate automatically on regular interval ground station antenna pointing ephemeris for the whole fleet .

Flight Dynamics Functions hosted in QUARTZ ++:

Basically, QUARTZ ++ inherits from the flight proven QUARTZ and COSMIC Flight Dynamics

Station keeping Maneuver planning activities:

The FDS is designed to cope with various station keeping strategies and provide FDS operator with a lot of flexibility for maneuver scheduling. One may plan as many station keeping maneuvers in advance based on regular or irregular station keeping cycles. The basic idea when planing a new maneuver is to append it to the current maneuver set. The effect on the orbit of all scheduled maneuvers are then automatically taken into account for other maneuvers planning or any flight dynamics activity that require orbit propagation. All orbital maneuver planning activities have the module architecture presented on the diagram below:



The selection maneuver module displays in its input GUI the list of maneuvers in the current maneuver set. This allows the user to select one of these maneuvers if the purpose of the planning is to update an existing maneuver rather than computing a new one. (Typically, once an new orbit has been determined one may want to adjust the parameter of an existing maneuver as a function of the updated orbit).

The theoretical delta-velocity computation module computes the theoretical orbital corrections that compensate orbital perturbations. Several types of station keeping maneuvers are available (a different activity is invoked for each kind of maneuver):

- Inclination control maneuvers: the software can compute both fixed cycle duration maneuvers and maneuvers that maximize the duration until the next orbital correction. Processing options are available to anticipate the eclipse seasons for spacecraft which cannot cope with inclination control during the eclipse season. The software may also compute EW compensation maneuvers that completely or partially compensate the predicted in-plane effect of NS maneuvers.
- Longitude eccentricity control maneuvers: Both fixed cycle duration maneuvers and maneuvers that maximize the cycle duration may be computed. The algorithm aims at minimizing the number of double East-West corrections in order to diminish the operational workload. The computation of E/W corrections anticipates the predicted effects of on-coming N/S maneuvers.

- Station acquisition and longitude re-allocation maneuvers: computation of E/W maneuvers for fine station acquisition at the end of LEOP, for station longitude change or positioning on graveyard orbit.

All these types of maneuvers are compatible with collocation strategies based on eccentricity-inclination separation.

The archive and telemetry data retrieval module aims at retrieving from the telemetry propellant values such as temperatures and pressures. It may also scan the maneuver archives in order to derive a predicted performance factor from the values computed for previous maneuvers.

The maneuver implementation module converts the theoretical velocity increment previously computed into maneuver start time and duration as a function of the propellant data. This algorithm involves numeric orbit integration through HI-FI propulsion models. The propulsion model is specific for each spacecraft. (For INTELSAT FDS version, MMS developed custom implementation modules that will operate INTELSAT's satellite fleet . These custom modules are INTELSAT proprietary property) . Once the maneuver is implemented, it will be saved in the maneuver database in a form that is independent of the spacecraft so that it can be taken into account in orbit propagation by all the generic flight dynamics algorithms of the FDS.

The last module is dedicated to constraints verification: this module automatically computes all the relevant sensor or orbital events that may not be compatible with the selected maneuver mode. Warning is issued in case of constraint violation.

Station Keeping maneuver reconstruction:

The maneuver reconstruction activity is invoked typically 24 hours after the end of a maneuver. This activity invokes the orbit determination modules (depicted later) in order to solve for the orbital effect of the maneuver. It retrieves the thruster actuation telemetry data in order to simulate the maneuver with the HI-FI propulsion model with the actual thruster actuation ON-Times. The fuel consumption during the maneuver is thus accurately estimated. The modeled delta-velocity is compared to the orbital effect estimated from orbit determination in order to derive the performance factor of the maneuver.

Transfer Planning:

QUARTZ ++ includes a complete set of algorithm for apogee/perigee burn planning during LEOP operations. The FDS can cope with classical Geo-synchronous Transfer, with Super-synchronous transfer as well as sub-synchronous transfer.

The computation and optimization of the transfer burn sequence during LEOP involves a numeric optimization algorithm: the software solves for the parameters of each maneuvers (right ascension, declination, start time and duration) in order to maximize either the Begin Of Life (BOL) mass, either the mass after the last transfer burn. It is possible to define constraints on the burn parameters (typically the attitude of the next burn can be fixed to the current attitude value) as well as constraint on the orbit after each burn. The software also allows sensitivity analysis to be performed: typically one can evaluate the sensitivity of the BOL mass with respect to uncertainty on the burn attitude or with respect to a burn delay.

For pre-launch transfer strategy design and mission analysis, transfer sequence design activity (including the design of back-up strategy in case of missed burn), launch window computation activity, dispersion analysis activity are hosted in QUARTZ ++.

Orbit determination:

Orbit determination is based on a Weighted Least Square algorithm. In addition to the orbit parameters, the filter is able to solve for different measurement and ground station biases as well as station keeping maneuvers. The covariance matrix associated to the solved for state vector is propagated: one can use the orbit determination software to evaluate the impact of measurement error or biases on the orbit determination accuracy. Smoothing algorithms are also available to pre-filter measurement bursts. A dedicated module allows the user to handle measurement data points manually.

Orbit propagator and Event predictions:

The FDS orbit propagator is based on a Runge-Kutta Butcher order 7/8 algorithm. The orbit is propagated through the scheduled maneuvers. The FDS generates various ephemeris products and predicts orbital events (eclipses, sun interference), predicts visibility or blinding for several kind of sensors, interference with other spacecraft on the geostationnary ring and computes ground station visibility and link budget.

Attitude determination and maneuvers:

For spin-stabilized spacecrafts, the FDS includes an attitude determination software. Attitude determination is based on a WLS algorithm which can solve for measurement and sensor wedging biases as well as solar torque components in addition to the attitude. Closed form methods are also available for filter initialization. Attitude maneuver planning activities are available for the control of the spin rate and the spin axis direction.

Relative motion prediction:

For collocated satellites on-station, prediction of the relative motion of the satellites in the cluster are performed. The relative motion can be plot and warning are raised if safety margins are violated.

Perspectives:

In a close future (begining of 2000), QUARTZ ++ will benefit from additional functions which aim at improving the level of automation for the control of large fleet of geo-stationary satellites. In particular the orbit and attitude determination tasks will be automated. For that purpose, orbit and attitude measurements will be automatically and continuously processed by Kalman filter estimation algorithms.

An interface with TIMELINE EXECUTER S/W (operation scheduling software developed by MMS) will feature advanced functions for automatic scheduling of station keeping maneuvers. Scheduled maneuvers for the whole fleet will be presented on the TIMELINE display for a typically 2 months period. On user request the maneuver schedule will be updated automatically and new maneuvers computed if necessary. Specific flight dynamics tasks will be initiated directly from the maneuver schedule.

In addition to an operational use for geosynchronous transfer and station keeping operations, the openness and the configuration capacities of QUARTZ ++ makes its application to Low Earth Orbit satellites control possible. This domain constitutes one of the evolution direction of MMS QUARTZ ++ product.

Acknowledgment:

INTELSAT FDC will be the first industrial application of QUARTZ ++. INTELSAT owns and operates a global communications satellite system. With 1998 revenues of over $1 billion, the INTELSAT system provides voice/data, video and Internet services to over 200 countries and territories.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 133-139
ISSN 0100-7386
Printed in Brazil

Automated Flight Dynamics Product Generation for the EOS AM-1 Spacecraft

Carla Matusow
Robert Wiegand
National Aeronautics and Space Administration
Goddard Space Flight Center
Code 583 Greenbelt, MD 20771 USA
Carla.Matusow@gsfc.nasa.gov
Robert.Wiegand@gsfc.nasa.gov

Abstract

Because of the complexity of the AM-1 spacecraft, the mission operations center requires more than 80 distinct flight dynamics products (reports). To create these products, the flight operations team will use a variety of modified commercial and National Aeronautics and Space Administration (NASA) developed flight dynamics software applications. Unfortunately, this means routine product generation requires flight dynamics expertise, requires skills in using each software application, takes several hours of operator interaction, and has many opportunities for user errors.

To address these issues, we (the flight dynamics team) developed automation software, called AutoProducts, which provides all the necessary coordination and communication among the various flight dynamics applications. AutoProducts autonomously retrieves files; sequences, initializes, and executes applications; and delivers the final products to the appropriate customers. This eliminates the need for flight dynamics expertise and knowledge of each application for routine product generation. Also, it virtually eliminates the potential for error and routine product generation needs no human-computer interaction.

Although AutoProducts required a significant effort to develop because of the complexity of the interfaces involved, its use will provide significant time and cost savings through reduced operator time and maximum product reliability. User satisfaction is significantly improved with AutoProducts and flight dynamics experts have more time to perform valuable analysis work. In addition, most of the AutoProducts code can be easily reused for future missions.

Keywords: *automation, autonomous operations, commercial-off-the-shelf, flight dynamics.*

Introduction

As part of National Aeronautics and Space Administration's (NASA's) Earth Science Enterprise, the Earth Observing System (EOS) AM-1 spacecraft is designed to monitor long-term global environmental changes. Using the spacecraft flight operations system, EOS Mission Operations System (EMOS), the flight operators send commands and receive telemetry from AM-1 (Figure 1). The AM-1 Flight Dynamics System (FDS) provides important information to EMOS, which is used to create spacecraft commands and plan mission events. The flight dynamics information is packaged into more than 80 distinct products (reports). Specifically, the operators use FDS to:

- monitor spacecraft attitude and orbit in real-time
- provide spacecraft commanding information
- support anomaly resolution for spacecraft navigation, maneuver, and attitude systems
- perform spacecraft maneuver planning
- analyze spacecraft on-board orbit computations
- predict potential spacecraft communication times
- provide spacecraft science instrument planning aids

Because the mission-required flight dynamics information is so diverse, FDS consists of several integrated commercial and custom software applications. Without any automation software, FDS was extremely cumbersome, requiring flight dynamics expertise and knowledge of several software packages and hardware platforms. Routine product generation was a highly interactive process that took hours to complete and had the potential for many user errors.

Problem description

In an effort to reduce software development costs and shorten development time, the flight dynamics organization at NASA regularly evaluates the use of commercial software packages. Therefore, FDS incorporates Satellite Tool Kit (STK) by Analytical Graphics, Inc., FreeFlyer by AI Solutions, Inc., and MATLAB by The Math Works, Inc to meet the AM-1 requirements (Figure 2). However, all the

commercial software needed customization. Even after customizing the commercial software, we still needed to write some unique software to meet requirements specific to AM-1.

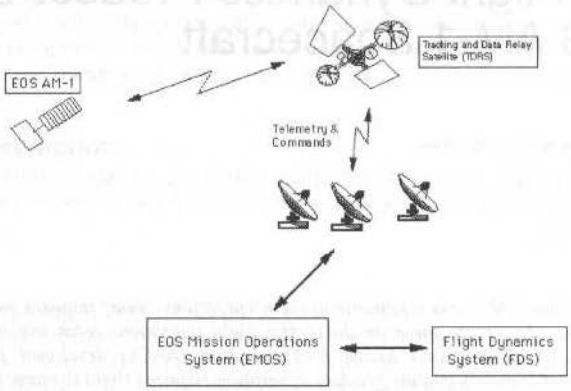


Fig. 1 AM-1 Flight Operations Ground System Components

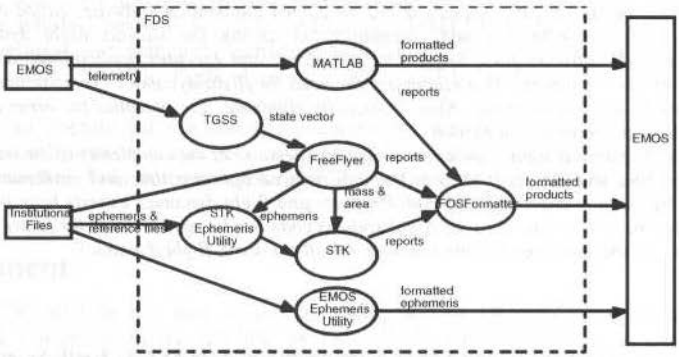


Fig. 2 FDS Architecture Diagram Without AutoProducts

The STK package allows users to compile custom software modules with the commercially available STK modules to expand its functionality. We wrote several custom modules (in C) to produce acquisition data, lunar beta angles, lunar eclipse times, solar terminator, solar eclipse, solar mid day, spacecraft shadow times, and minimum/maximum latitude times – AM-1 specific functionality that wasn't available in the commercial package. We use built-in functionality to generate fields of view, viewing times, orbital events, solar and lunar azimuth and elevation angles, and altitude reports. STK uses predicted ephemeris files generated by FreeFlyer to produce its reports.

The FreeFlyer package provides an object oriented scripting language to allow users to perform tasks. We wrote several complex FreeFlyer scripts to produce maneuver planning products. We use relatively simple FreeFlyer scripts for predicted ephemeris, state vectors, and geometrical event reports (e.g., Brouwer-Lyddane elements, solar beta angles, and local sun time). FreeFlyer uses institutional reference files and a state vector from the spacecraft telemetry to produce its reports.

The MATLAB package requires users to write scripts using MATLAB library routines to perform tasks. We wrote several custom MATLAB modules to create a variety of spacecraft attitude related products. Our customized MATLAB modules use telemetry files to create the attitude data and produce several products in the EMOS format.

AM-1 has on-board navigation software, called the Tracking and Data Relay Satellite (TDRS) On-board Navigation System (TONS). A copy of the TONS flight software, implemented on a Sun Microsystems workstation, as well as some supporting software, is used to analyze spacecraft on-board orbit computations and performance. This integrated set of software is called the TONS Ground Support System (TGSS). TGSS uses telemetry files to validate orbit vectors from the spacecraft. FreeFlyer needs this information to generate its reports.

We modified existing NASA-developed ephemeris file reformatting software (Ephemeris Utilities) to produce the specific formats required by STK and EMOS.

Each product has precise data format requirements. Because FreeFlyer and STK cannot meet the formats required by EMOS, we wrote a custom application using Perl, called FOSFormatter, to reformat reports into the formats EMOS requires. Nearly all products must be reformatted by FOSFormatter before being delivered to the customers.

Every day, the flight operations team is required to generate 30 different flight dynamics products. Under special circumstances, even more products must be generated. From the flight operators' perspective, generating the daily products is a time-consuming task. The instructions to generate these products take 18 pages! Integrating several software applications into the FDS raises several operational concerns:

- Routine product generation requires knowledge of multiple applications executing on different hardware platforms. Generating daily products requires knowledge of UNIX, Windows NT, TGSS, STK, FreeFlyer, MATLAB, FOSFormatter, and the AM-1 Ephemeris Utilities.
- Generating products is a highly interactive process requiring a user to interact with each application multiple times to generate each product.
- Routine product generation requires several hours to complete. Although each daily product can be generated in 5-40 minutes, the entire process takes 6-8 hours because each product must be generated individually and serially – the first product must be completed before the next product can be started.
- User interaction with each application introduces the potential for errors, since users are required to manually enter filenames and input parameters as well as sequence and execute applications. Even with 18 pages of detailed instructions for daily product generation, operators commonly make several mistakes.
- Generating products requires some level of flight dynamics expertise to determine appropriate inputs and sequencing. Operators need to understand the system fairly well in order for product generation to make sense and prevent critical mistakes.

During FDS development these issues became apparent when developers and flight dynamics analysts began generating sample products for testing purposes. It quickly became clear that using several different software applications was unreasonable and some sort of automation was necessary to make the system easier to use.

Solution Description

To address these issues, we developed a software application called AutoProducts. The purpose of AutoProducts is to capture operations procedures. AutoProducts performs routine product generation without human-computer interaction. In addition, it acts as a single graphical user interface for all the flight dynamics software applications to allow users to generate unique groups of non-routine products easily.

AutoProducts is executed from a single hardware platform and provides all necessary coordination and communication among the various flight dynamics software applications. AutoProducts autonomously retrieves necessary files, sequences and executes applications (on the same platform and on other hardware platforms) with correct input parameters, and delivers the final flight dynamics products to the appropriate customers. Although AutoProducts will normally generate pre-programmed sets of routine products, its graphical user interface allows for configuration of customized and one-of-a-kind products. Additionally, AutoProducts has been designed as a mission-independent tool, and can be reconfigured to support other missions or incorporate new flight dynamics software applications. AutoProducts is capable of generating the appropriate products automatically at pre-determined time intervals for the life of the mission.

AutoProducts Framework

AutoProducts is an extensible framework. The key to understanding AutoProducts is the concept of Actions (Figure 3). An Action holds data that parameterizes a task that AutoProducts carrys out. Support for various kinds of Actions are implemented in modules that are loaded into AutoProducts at run time. For a particular kind of Action, the Action Execution module verifies and performs the Action. The Action Editor module displays and allows the user to modify the Action's data.

Actions fit nicely into the object oriented programming model. An Action has a Type, which defines how it implements the Action interface. Every Action understands the *validate*, *execute*, *dependencies*, and *edit* messages. The *validate* message requests a consistency check on the Action's data. The *execute* message carries out the Action with its current data (configuration). The *dependencies* message requests

a list of the Actions that this Action is dependent upon. When AutoProducts runs interactively, the *edit* message presents an Action Editor to the user. Through the Action Editor, users customize the Action's data.

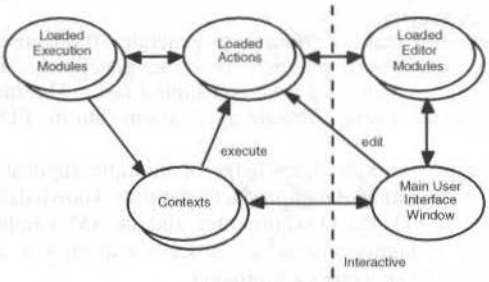


Fig. 3 AutoProducts Framework

An example of a kind of Action is a List Action which groups and orders other Actions. Its data consists of a sequence of Actions and a flag for each indicating whether or not it is currently enabled. When a List is executed, it executes each of its enabled elements. Users enable or disable Actions within the List and add or remove Actions to the List through the List Editor.

An Action executes within a Context. The Context preserves state across Action executions. Actions communicate with each other by manipulating the Context. The Action may have effects outside of AutoProducts (e.g. running another application), but coordinating these effects is handled through the Context. The Context maintains a stack of currently executing Actions. Contexts allow AutoProducts to perform Actions concurrently. The user interface allows creating a new Context and selecting the Context in which Actions will execute.

Actions and Types are maintained in a Registry (Figure 4). Loading an Action Execution module defines a new Type. The Type maps the *validate*, *execute*, *dependencies*, and *edit* messages to functions in the loaded modules. Actions (instances of Types) are loaded from files. The Registry verifies the Actions it loads with the *validate* and *dependencies* messages. Actions are also verified when the user attempts to commit an update (made with an Action Editor) to the Registry.

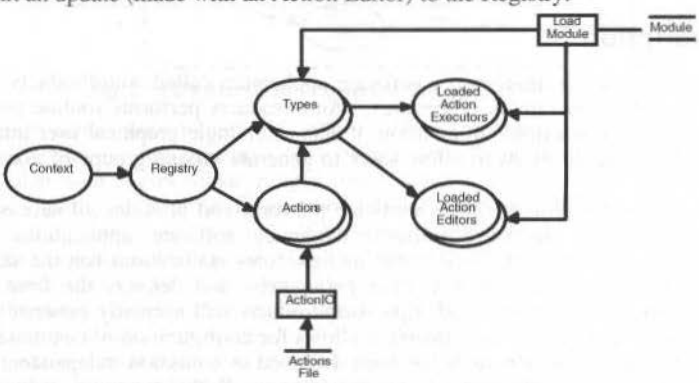


Fig. 4 AutoProducts Actions and Types

Each Context has its own Registry of Actions and Types. This is necessary since Actions may change in one Context (due to user interactions) while another Context is active. For Actions to safely execute concurrently, the Action execution module must maintain state in the Context and use the Context-unique identifier in external interactions. However, AutoProducts cannot guarantee that the application it is interacting with can perform the requested tasks in parallel.

Some Types of Actions have general application. A List Action groups a set of Actions in a particular sequence. A Test Action executes another Action and compares the results to the expected results. An Eval Action provides a hook to the implementation language.

The following are other key parts of the framework. The Messaging module implements the protocol for sending and receiving messages within AutoProducts. The Context service creates and commands existing Contexts (e.g. tells a Context to execute an Action). The Scheduling service schedules Actions for execution in a particular Context at a later time. The Reporting module collects and distributes status, warning, and error messages.

The AutoProducts framework allows modules to be plugged in when necessary. Adding support for a new Type of Action is simply a matter of implementing the Action interface. This might extend AutoProducts capabilities by interfacing with another software application. Implementing an Action Editor would support interactive operation. Different Action Editors, with more or less flexibility, can be plugged in for different users. Modules that do not directly support Actions can be loaded as well, for example, the module that provides support for AM-1 naming conventions.

When run interactively, AutoProducts presents a Perl/Tk based user interface. Figure 5 shows the main user interface window that presents the list of available Actions. When the user requests to edit an Action, an Editor for that Action is opened (Figure 6). That Editor may open other windows (Figure 7). All of the Editors in the user interface populate a window having a common shell for consistency.

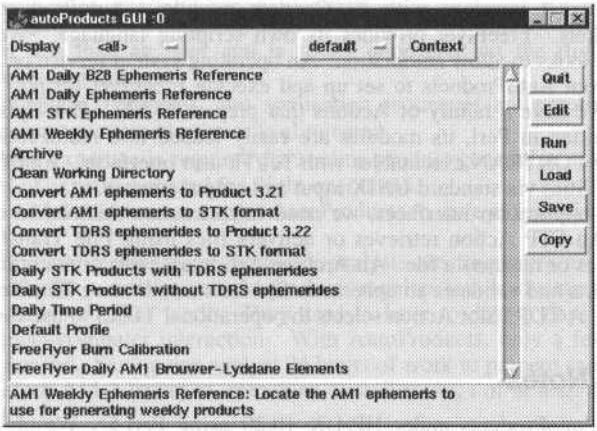


Fig. 5 AutoProducts Main User Interface Window With AM-1 Actions Loaded

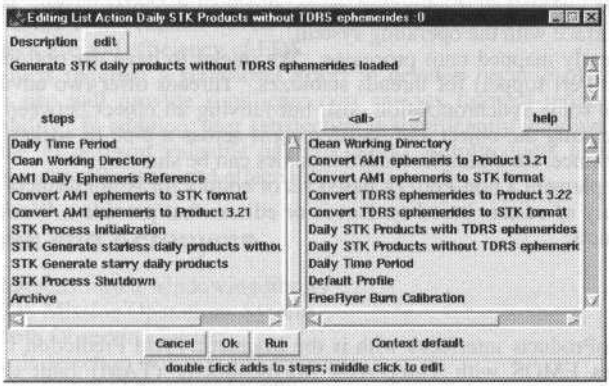


Fig. 6 Sample List Action Editor Window

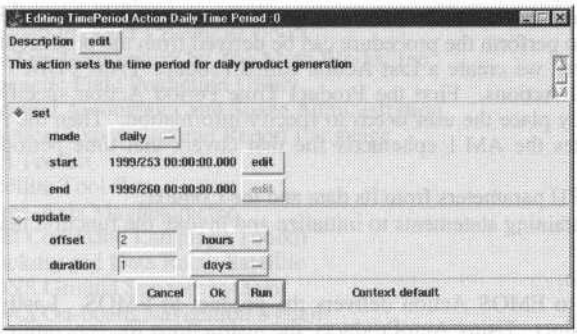


Fig. 7 Sample Action Editor Window

AM-1 Specific Implementation

To support the AM-1 mission, we developed a number of Action modules. We expect that many of these modules will be useful in future AutoProducts applications.

Supporting an external tool is dependent on the interface it provides. We were fortunate to have source code for many of the tools we need to support, simple interfaces to others (e.g., many UNIX utilities), and scripting languages and/or good vendor support for others. Trying to get events into an application that only provided a graphical interface would be a problem.

For AM-1, AutoProducts needs to interface with several software applications. We created Actions to work with each application: STK, FreeFlyer, MATLAB, FOSFormatter, and the Ephemeris Utilities. STK provides an external interface with its Connect module. AutoProducts communicates with STK/Connect via sockets. FreeFlyer provides its own scripting language. We annotated the AM-1 FreeFlyer scripts with hooks to allow automation. An operating system service on the FreeFlyer platform receives commands from AutoProducts to set up and execute FreeFlyer scripts. MATLAB also has a scripting language. We have a family of Actions that prepare MATLAB scripts for execution. Since FOSFormatter was written in Perl, its modules are easily loaded into AutoProducts at run-time. The Ephemeris Utilities are FORTRAN executables with Tcl/Tk user interfaces. AutoProducts communicates with the Ephemeris Utilities via standard UNIX input and output streams.

In addition to the application interfaces, we need infrastructure Actions to coordinate the product generation process. An FTP Action retrieves or delivers files using File Transfer Protocol (FTP). A Reference Action moves or renames a file. An Archive Action accesses or maintains the product archive. An Ephref Action selects and validates an ephemeris file for a satellite. A TimePeriod Action selects the timespan for products. A TDRSSlot Action selects the operational TDRS to use for product generation.

Implementation Notes

We implemented AutoProducts under HP-UX 10.20 using Perl 5. We chose HP-UX because it supports the largest subset of the software applications AutoProducts needs to coordinate. Perl 5 has strong module support and makes it easy to dynamically load modules. Also, Perl 5 has an extensive library and a good interface with the operating system.

Contexts are currently mapped onto processes. We intend to have the option of mapping Contexts onto threads when the Perl support for threads stabilizes. Threads offer two advantages. First, threads will be faster: there is some synchronization cost, but moving an object between queues in memory is much faster than serializing the object and transferring it across a pipe or socket. Second, threads will reduce resource usage since, for example, loaded modules can be shared.

It is possible to implement a non-Perl/Tk based set of editors for AutoProducts to load. For example, we could build a family of editors based on *curses* or editors that provide a World Wide Web (WWW) interface to AutoProducts.

Example

One tool that AutoProducts interfaces with is the Magnetic Field Prediction Utility (MFPU). Upon request, FDS provides EMOS with Three-Axis Magnetometer (TAM) fault detection isolation and recovery (FDIR) predict tables. These tables (created by MFPU) are required for on-board fault detection during attitude maneuvers. MFPU is implemented as a MATLAB application with its own user interface. The procedure to produce and deliver this table requires several pages of instructions. However, all the information necessary to perform the procedure can be derived from the product start time.

Within AutoProducts, we create a List Action called Produce TAM FDIR Predict Table. This List Action contains several Actions. First the Product Time Period Action specifies the start time of the product. This is the only place the user needs to specify information. Then the AM-1 Period Ephemeris Reference Action locates the AM 1 ephemeris file that covers that time period. Next, the MATLAB MFPU Action:

- derives the MFPU parameters from its data and the Context;
- builds a file containing statements to initialize and invoke the function that generates the product; and
- executes that file.

The FTP Products to EMOS Action delivers the product to EMOS. Lastly, the Archive Products Action archives the product. Using AutoProducts, the instructions for generating the TAM FDIR Predict Table are reduced to a few mouse clicks and selecting a start time.

Conclusion

AutoProducts greatly reduces many of the concerns associated with the flight dynamics product generation:

- Users can be trained to use a single application and graphical user interface to generate products. Groups of non-routine products can be generated with a few minutes of human-computer interaction using a single user interface. Users require knowledge of HP-UX and AutoProducts only.
- Routine product generation is performed autonomously. No interaction is necessary, allowing the operator to do other tasks during product generation.
- Routine product generation processing time has been significantly reduced. Daily product generation takes less than an hour and is solely dependent on the flight dynamics software applications' processing time.
- Since input parameters, filenames, and application sequences are preprogrammed for routine product generation, user interaction error is eliminated.
- No flight dynamics expertise is required for routine product generation. Autonomous operation reduces the level of expertise needed for flight operations.

Over the life of the AM-1 mission we expect AutoProducts to save at least 6 hours of operator time every day.

While EMOS is still being tested, flight dynamics analysts are frequently asked to provide sample products. With AutoProducts, generating sample products requires less lead-time and has a significantly smaller impact on the analysts' other work. They used to have to devote days to generating sample products – hours of human-computer interaction. With AutoProducts, only a few minutes of human-computer interaction is needed. Saving one analyst 24 hours of work to prepare sample products for each of 20 remaining tests before AM-1 launches, means an overall savings of at least 480 hours of analysts' time by using AutoProducts.

In addition, training users to use AutoProducts instead of all the flight dynamics software applications is easier. It requires less time and users gain a better understanding of the overall system.

Since users began using AutoProducts (even with limited capabilities), we have received positive feedback about the ease-of use and efficiency of FDS.

Although AutoProducts required a significant effort to develop because of the complexity of the interfaces involved, its use will provide significant cost savings through reduced operator time and maximum product reliability. In addition, user satisfaction is significantly improved and flight dynamics experts have more time to perform valuable analysis work. AutoProducts helps both analysts and developers do their jobs more efficiently and effectively.

Glossary of Terms and Acronyms

curses	character based windowing library
EMOS	EOS Mission Operations System
EOS	Earth Observing System
FDIR	fault detection isolation and recovery
FDS	AM-1 Flight Dynamics System
FORTRAN	formula translation, a programming language
FTP	file transfer protocol
HP-UX	Hewlett Packard's flavor of UNIX
MATLAB	MATrix LABoratory by The Math Works, Inc.
MFPU	Magnetic Field Prediction Utility
NASA	National Aeronautics and Space Administration
Perl	Practical Expression and Report Language
Perl/Tk	Perl Toolkit
STK	Satellite Tool Kit
TAM	Three-Axis Magnetometer
Tcl/Tk	Tool Command Language Toolkit
TDRS	Tracking and Data Relay Satellite
TGSS	TONS Ground Support System
TONS	TDRS On-board Navigation System
UNIX	a multi-user operating system
Windows NT	a Microsoft operating system

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 140-148

ISSN 0100-7386
Printed in Brazil

ODS: An Efficient Flight Dynamics System for Satellite Station Keeping (ODS: Orbital Dynamics Software)

Pascal Wuszko
CS – CISI
ZI du Palays, 13 rue Villet
31029 Toulouse Cedex France
pascal.wuszko@cisi.fr

Helene Gautier
Dominique Delmas
CNES
18, avenue Edouard Belin
31401 Toulouse Cedex 4 France
helene.gautier@cnes.fr

Abstract

ODS (Orbital Dynamics Software) is dedicated to the station keeping of 3-axis stabilized geostationary satellites. It is a set of space dynamics modules integrated into the very powerful Man Machine Interface (MMI) MERCATOR II.
This software is a safe solution to the automatic chaining of the space dynamics calculations and operations.
It is developed by CNES (the French space agency) with CS CISI (a subsidiary branch of the "Communications and Systems" group) as subcontractor.
OSD is a new generation of ground control software used by CNES (TELECOM2), ARABSAT, the Swedish Space Corporation (SIRIUS 2), Shinawatra (THAICOM 3 colocated with THAICOM 2) ... ODS may be installed as a partial or complete solution for the upgrade of old ground control centers (Example: TDF). Today, 10 satellites are controlled from 6 different sites.
Keywords: *Station, Keeping, Geostationary 3-axis Stabilized Satellites, Satellite Control centers.*

General Information

The main principle of ODS is the complete automatic chaining of the space dynamics calculations during the whole operational life of the 3-axis stabilized satellites on the geostationary orbit. That means ODS is able to

- Determine the centered adapted parameters of the orbits;
- Compute the maneuvers,
- Generate the telecommands,
- Exploit the telemetry and the ranging data.

ODS is integrated into a Man Machine Interface provided by MERCATOR II. All the space dynamics modules benefit from this MMI which is able to monitor, mange and control:

- The parameters setting,
- The execution of the software,
- The accuracy of the results.

A graphical tool (a reference software in CNES flight dynamics activities) is used to plot the station keeping graphs exploited by the operators to instantaneously understand the orbit control of the satellite or to help the writing of the flight operation reports.

MERCATOR II is also used by CNES during the LEOP of the geostationary satellites. This MMI has been validated in operation mode by 27 LEOP since 1989.

When starting ODS, the operator starts MERCATOR II and gets a main menu (See Fig. 4) where all the functions are reachable through a simple pushbutton: a simple click on the mouse and a space dynamics computation can be run.

The powerful MMI, the high skills of CNES and CS CISI teams in term of geostationary station keeping, the ability of the software to summarize the main results of the computation can be run.

The powerful MMI, the high skills of CNES and CS CISI teams in term of geostationary station keeping, the ability of the software to summarize the main results of the computations make ODS one of the best systems to be used.

The main modules (orbit determination, celestial events and maneuvers calculation) are directly applicable to whatever 3-axis stabilized geostationnary spacecrafts.

All specific attitude control, telecommand and telemetry modules can be developed using the CNES space dynamics software patrimony.

Thanks to the perfect automatic chaining of the operations and to the ability of ODS to summarize the main information, only 1 engineer at a time is required to operate the ODS and a team of 2 engineers are enough to maintain the satellite in station keeping. The money saving are not negligible after a few years exploitations with respect to the old systems.

ODS is not an off-the-shell software. However it can be adapted, validated and released in less than 6 months and sometimes in only a few weeks. ODS is a low cost solution. It can be adapted to the mission of a satellite with a minimal development effort. Moreover, because the satellite mission may change during its life (longitude window, eccentricity circle, colocation with other satellites), ODS has been designed to be easily evolutionary. In most of the cases, only a modification of the data base is necessary. The maintenance of the software is another strong point of ODS.

One could ask what is necessary to do to install ODS in a new Satellite Center or to install it in a old SCC. The next paragraph shows that ODS is a solution for the brand new SCC and also a solution for the upgrade (or retrofit) of the old SCC.

Installation and Configuration of ODS

ODS is based on a set configuration files written before the release of the software. These files describe the arrangement of the space dynamics modules, the arrangement of the data base, of the ranging files, of the retrieval files used in operation and of the temporary files.

Example: a configuration file defines all topics of the main menu (See Fig. 4) and links the modules together in a given topic (the maneuver calculation module is accessible when clicking Maneuver topic in the main menu. See Fig. 5).

A specific shell has been developed to automatically configure these files by a few environment variables before the release.

The installation of ODS consists in copying the software from a storage support (CD, DAT, ...) to a dedicated space of 100 Mo. RPM (Red hat Package Manager) can also be used, providing a most up to date and efficient solution for installation.

The computer must be a SUN with a Solaris System or a PC with a LINUX system.

The environment of ODS is automatically set using the configuration files when starting ODS. The procedure is very simple since only a command has to be typed.

Once ODS is installed and configured, it can easily be used by the operators thanks to MERCATOR II. All the functions are described in the following paragraph.

ODS: an easy and powerful MMI

Inputs Management:

Each space dynamics module of ODS has inputs. They are extracted from a data base entirely managed by the MMI. The operator can or cannot modify them according to the ODS manager choices.

Each input data modified or set by the operator is controlled by the MMI. Only specified values are authorized. The controls are defined before the ODS release by the ODS manager and by the mission analysis.

Outputs Management:

After executing a space dynamics module, the operator gets outputs. He can access them using the MMI and store them in the data base. They are controlled by the modules. It is the principle of the automatic chaining of the operations. No "manual" operation is needed.

The operator can use a synthesis page to broadcast the main results on several screens of the computer network. He can use the history function to save the results and analyze them over several cycles.

He can run the integrated graphical tool of ODS to plot graphs. All the graphs procedures by ODS may be saved under a post-script format and included in a cycle report.

Easy Chaining of the Operations. OSD is Easy to Use:

The complete chaining of the operations is described by a User's Manual delivered with ODS.

After opening the main menu of the MMI, the operator can access the whole modules by a simple click on pushbuttons.

It is then very easy to run the modules because in most cases no input has to be set by the operator.

Only a few days are necessary to train operators how to chain the modules.

During the parameter setting, the execution and the checking of the results, a log-book is open. Messages are displayed to give information about the controls, to help the operators or to prevent all the problems.

History

An history function is implemented in ODS. Its function is to allow the operator to store in files whatever data he wants and therefore to help the analysis of the flight dynamics parameters of the orbit, of the satellite and of the ground antennas during the whole life of the satellite in order to improve the data base.

Example: The evolution of the ratio between the tangential and the normal component of an inclination maneuver versus the local satellite time leads no the full management of the cross-coupling phenomenon.

An external graphical tool (EXCELL) can be used as well as the ODS's one to plot the evolution of the interesting parameters over the operational life of the satellite.

Retrieval Files

Each module can be run using the current state of the data base or using a set of inputs prepared and stored during a previous execution of the module. This is the principle of the retrieval file.

After each execution the operator can store the inputs and outputs of a module in a specific retrieval file. This file is then used to check the operations (tracability of any computation) or to train the novice operators how to use ODS.

In most of the installations two ODS exist. The training and the operational ODS share the same space dynamics modules, but have two different sets of configuration files and two different data bases.

The novice operators can train on the training ODS while the operations are done on the operational ODS without any risk.

Broadcasting

Using the Broadcasting function, the operator can "hardcopy" a page on its screen or a printer. This special page summarizes the main results of an execution. It can be broadcast on several screens (and several computers according to the mount of the network).

An operator and several analysts can work together at the same moment with ODS.

But thanks to a lock-system, only one person has the right to modify the data base.

ODS, Satellite Control Center and Satellite

The information is always controlled by the MMI. The operator accesses to the data base using the MMI. The telecommands (or the data needed by the procedures) are available on the broadcasting function of the MMI.

The operator can access to the telemetry and ranging files using the MMI.

The operator can access to the data base of other control systems using the MMI (in case of colocated satellites).

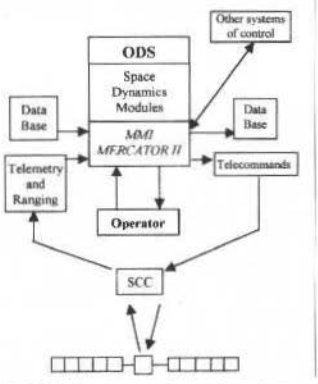


Fig. 1 ODS in the control system of the satellite

Start of the cycle, Entry Point

The simplest way to understand the ease to use an ODS is to present the chaining of the operations through a 2 week East-West and North-South cycle.

At the very beginning of life of the satellite, it is necessary to initialize the data base with data coming from the LEOP (initial orbit, mass, fuel and oxidizer pressures, temperatures in the tanks ...).

Once these data have been entered, the data base of the satellite will be entirely managed by ODS during its whole life.

The principle of the station keeping is to define an East/West cycle which duration is an integer number of weeks. The corresponding North/South control is based on a cycle multiple of the East/West one.

The computation of the maneuvers is rythmed by the entry points (EP) of the East/West cycle.

At the very beginning of life of the satellite, it is possible to initialize the data base at any Entry Point. The software will then chain all the station keeping calculations from an entry point to the next one.

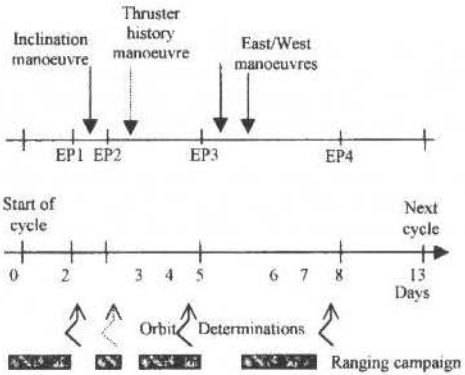


Fig. 2 Cycle schedule and operations

At EP1, after an orbit determination, the inclination maneuver is calculated and then performed.

At EP2 (a few hours after the inclination maneuver), a "thruster history" maneuver may be necessary in order to avoid the exit of the longitude window when the cross coupling of the inclination maneuver is very different from the foreseen one.

At EP3, the inclination maneuver is calibrated, the orbit is determined and an East/West maneuver or a couple of them is calculated to control both the eccentricity and the longitude.

At EP4, the operator calibrates the East/West maneuver and prepares the next cycles.

The ranging is stopped until the end of the cycle.

The colocation function of ODS is used whenever a maneuver has to performed. two (or more) satellites can be monitored in the same station keeping longitude. In that case ODS is able to get the orbit bulletin of all satellites to be colocated thanks to the MMI configuration files.

Entry Point 1

For a two week North/South cycle this entry point is used to calculate an inclination maneuver. The software can take into account a free drift period without any control of the secular inclination drift. In that case this entry point is mute.

Before calculating the inclination maneuver to be performed on the third day of the cycle, an orbit determination is necessary. 48 hours measurements are usually performed from the start of the cycle.

ROMANCE, the orbit determination module, is run. This module has an analytical model for the orbit extrapolation. It is able to determine the components of 5 maneuvers using angular, ranging, turn around or range rate measurements. Even the smallest speed increments are taken into account and determined (for example the wheel unloading thruster activity.

In output the operator gets an orbit bulletin. He updates the data base with the refreshed orbit by a simple click on the mouse and a confirmation. All the orbits got from ROMANCE are stored by the history function in a dedicated file.

This orbit is now the input of the maneuver computation module. This is the principle of the automatic chaining: *The outputs of a module become the inputs of the next one.*

The graphical tool can be used. It helps the engineers analyzing quickly the quality of the orbit determination. The residuals of the last Gauss-Newton algorithm loop are displayed.

ROMANCE is able to determine the 6 orbital parameters of the bulletin, the 3 components of up to 5 maneuvers, the antenna biases and the solar radiation pressure coefficient over mass (CpS/m) at the date of the orbit.

Taking into account this new orbit (and as often as necessary), a prediction module, GEOPRED, is run in order to compute the celestial, satellite and antenna events. GEOPRED may be run on the long term with a "frozen" orbit (constant centered adapted parameters) for example to predict the Sun eclipses by the Moon. It may be run on the short term with a fine orbit extrapolation for the exact forecast of the events.

The following events are calculated:

- Antenna-Sun interferences,
- Moon and Sun eclipses,
- InfraRed Earth Sensor blinding,
- Visibility of the Sun Sensors ...

Taking into account the new orbit, the inclination maneuver is calculated using the MANORB module. The principle of the module, for the normal control, is to control the secular perturbation on the inclination mainly due to the luni-solar potential. MANORB calculates the inclination drift over the 6 next months and estimates the necessary speed increments to be performed in order to keep the satellite in an inclination circle (the center of the circle is modifiable according to the mission of the satellite). This function is important since it is a way to separate colocated satellites (moreover eccentricity and longitude separation strategies are possible).

MANORB is used for the computation of the East/West maneuvers as we will see it in the next paragraphs. At Entry Point 1, MANORB computes the inclination maneuver and predicts the next East/West maneuvers.

In input, the operator can ask the systematic use of the Sun Sensors during the maneuvers or allow the use of the gyros. MANORB writes in its log-book which Sun Sensor or gyro is used to perform the maneuver (see Fig. 5)

When the Sun Sensors are used, a de optimization of the time maneuver is sometimes necessary in order to control the attitude of the satellite. MANORB is able to do it and estimates the over cost due to this de optimization.

After each maneuver computation a module dedicated to the colocation may be run. It is a mean to check that colocated satellites are correctly separated. If a close proximity occurs the maneuver plan of the maneuvering satellite is modified.

After each maneuver computation, the operator can ask for the plot of a graph showing the evolution of the orbital adapted parameters (See Fig. 7) over two or several cycles.

The true longitude, the eccentricity, the inclination, the drift and the mean longitude are displayed. It is a way to get a quick and easy glimpse of the satellite behavior.

This graph is the same as that of the extrapolation module (PREDORB). PREDORB is run for special operations when the orbital engineer has to compute "manually" a maneuver. It is able to extrapolate the orbit taking into account up to 20 maneuvers. The extrapolation is a numerical model; the smallest orbit perturbations are included in the algorithm.

An optional daily activity (in m/s/day) can be set (as well as for the MANORB module) to take into account an abnormal thruster activity due to important wheel unloadings for example.

After computing the normal maneuver, using the speed increments stored in the data base the operator, the telecommand session is started.

On the one hand a few modules are dedicated to the attitude control. For example, the questions is about the calculation of the Sun direction in the satellite reference frame or about the prediction of the Sun movement in the sensors.

Using the CNES software patrimony it is easy to develop whatever space dynamics module for the attitude control. So all the 3-axis stabilized geostationary satellites could be controlled by an ODS.

On the other hand, the maneuver page has to be produced in order to provide the data required by the procedures of the control center before performing the maneuver.

Two kinds of maneuver pages have been developed so far. They both use the performances of the thrusters calculated from the telemetry (pressures in the tanks and thrust models). A special module produces the steady state thrust and flow rate of each thruster. The operator stores these data in the base. They will be used by the maneuver page modules.

A set of hexadecimal translating modules are used in a few ODS. The principle is to convert the speed increments calculates by MANORB or PREDORB into hexadecimal value.

Specific modules could be developed for any kind of platform in order to work with any control center.

In all the cases the thrust profile (from ADCS) is implemented in the modules. The maneuver duration, the start date, the end date are calculated.

The wheel unloading to be performed before the station keeping maneuvers is computed from the on-board measured torques.

The main results of the maneuver page calculation are stored in the data base (maneuver duration, speed increments ...). They will be used at the following Entry Point for the orbit determination or the maneuver calibration.

Entry Point 2

Just after performing the inclination maneuver, the thruster history results are available in the telemetry. ODS can read the decommuted telemetry from the control center, determines the speed increments and the consumption.

The main objective of the thruster history exploitation is to compute the propellant remaining in the tanks and, of course, the mass of the satellite.

After executing the modules, these masses are updated in the data base.

According to the tanks and thrusters configuration, a thermodynamic model has been developed. It is used to check the calculation of the masses.

Another objective of the thruster history is to calibrate the performed maneuver by comparing the commanded (from the maneuver page) with the determined speed increments. This information will be used for the computation of the next maneuvers when the calibration by an orbital mean is not available.

The cross coupling associated with the inclination maneuver might lead to a longitude window exit. To avoid it, MANORB is run to Entry Point 2 using the determined speed increments in order to calculate a possible emergency East/West maneuver performed 6 hours after the inclination maneuver.

If necessary the telecommand session session is started again. In most of the cases, because the cross-coupling is a well known perturbation, this maneuver is not performed. MANORB is able to anticipate a tabulated cross-coupling. It has calculated a longitude rendezvous after the inclination maneuver of the current cycle with the East/West maneuver performed at Entry Point 3 of the previous cycle.

Using ROMANCE and its ability to determine the orbits "through the continuous maneuvers", the operator can get a new orbit at the date of the North maneuver with 48 hours before the North maneuver and only 6 or 8 hours after.

This orbit, stored in the data base, will be used for an orbital calibration of the maneuver. The orbit got from MANORB at Entry Point 1 dated before the maneuver and the determined orbit enable the calibration. The result of the calibration is historized. An analysis of the evolution of the calibration coefficient of the thrusters. These coefficients, stored in the data base, are used for the calculation of the next maneuver.

Entry Point 3

Using 24 or 48 hours measurements after performing the inclination maneuver (depending on the ranging precision and of the flight dynamics skills of the operators), a new orbit is got with ROMANCE and stored in the data base. A finest calibration of the inclination maneuver may be done.

Entry Point 3 is dedicated to the control of the longitude, of the drift and the eccentricity.

The longitude and drift controls are done by tangential speed increments MANORB is able to take into account a radial coupling and, as well as for the inclination maneuver, time constraints forbidding the performing of maneuvers during dead-bands.

The control of the eccentricity is more sophisticated. In a few cases the radius of the natural eccentricity circle is smaller than the control circle. Nothing special has to be performed and no shift of the time of the longitude maneuver has to be done. In most of the cases it is necessary to control the eccentricity because the control radius is smaller than the natural radius or because the satellites.

In these cases, MANORB first uses a single burn strategy by shifting the time of the longitude maneuver and, if it is not sufficient, switches to a two-burn strategy. MANORB is able to calculate 1 or 2 tangential speed increments (with a possible radial coupling) and to shift the time of the longitude maneuver in order to control the drift, the longitude and the eccentricity with the smallest over-cost.

The center and the radius of the eccentricity circle are in input of MANORB. The best circle, determined by the mission analysis, may be selected in order to separate colocated satellites.

The telecommand session is started for each East/West maneuver in order to get updated masses and to refresh the thruster performances before the next maneuvers.

An orbit determination and a calibration are executed after each maneuver using the ability of ROMANCE to go "through the maneuvers".

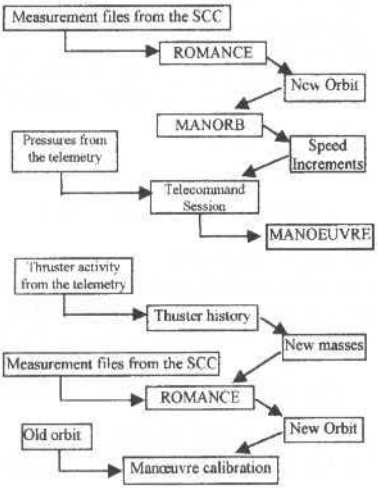


Fig. 3 Chaining of the operations before and after a maneuver

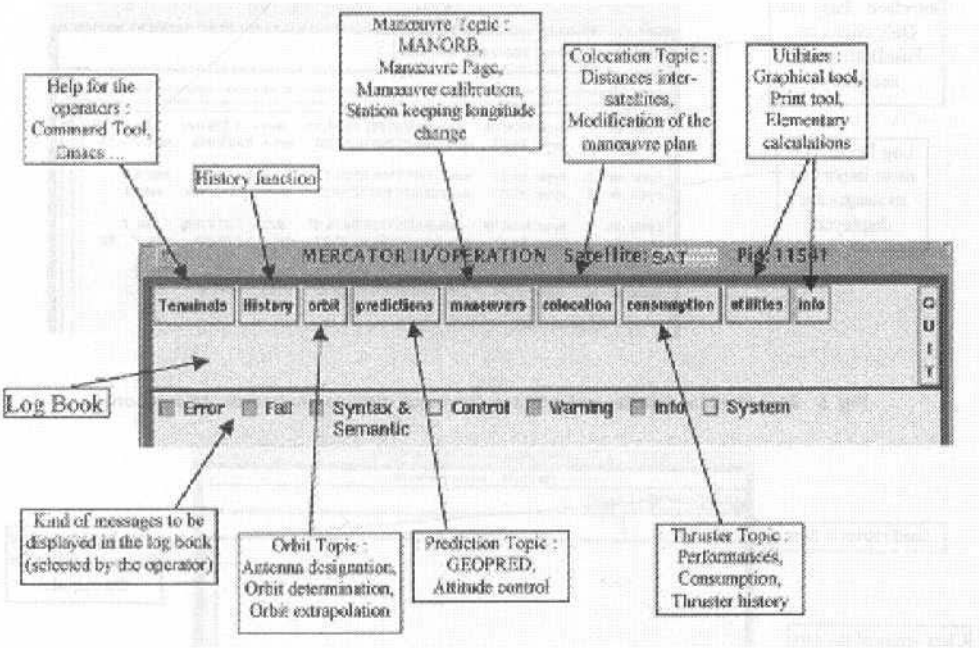


Fig. 4 Ergonomic Main Menu of ODS (MERCATOR II)

Entry Point 4

The last orbit determination is started 48 hours after the last tangential maneuver.
The new orbit is used to calibrate the last tangential maneuver.

MANORB is run only to prepare the next cycle. If an inclination maneuver has to be performed on next cycle, the Entry Point upgraded to 1.

If no inclination maneuver is foreseen, the Entry Point is upgraded to 3 and only tangential maneuvers will be performed during the next cycle.

A free drift period is command when the inclination maneuver should take place at a time where the Sun Sensors cannot be used and because the use of gyros is not allowed. This case is foreseen by MANORB: while the previous cycles it has given to the inclination vector a direction such that this vector will cross the center of the inclination circle just at the middle of the free drift period. It is a way to automatically avoid the exit of the inclination window with the smallest cost.

Conclusions

ODS is a highly adaptable tool dedicated to the station keeping of 3-axis stabilized geostationary satellites.

The Man Machine Interface, the main space dynamics modules such as MANORB, ROMANCE, GEOPRED the antenna designation module and the graphical tools are generic software.

The specific technological modules dedicated to the attitude control, to the telemetry exploitation or to the decommutation of the ranging may be developed within a few months using the CNES software patrimony.

The main evolution of ODS for 1999 and the next years are the integration of the plasmic propulsion. It will lead to a new ODS generation/

An improved, more ergonomic MMI will support the new complete chaining of the operations.

The on-line help will soon be introduced in order to give to the operators an again casier way to use ODS.

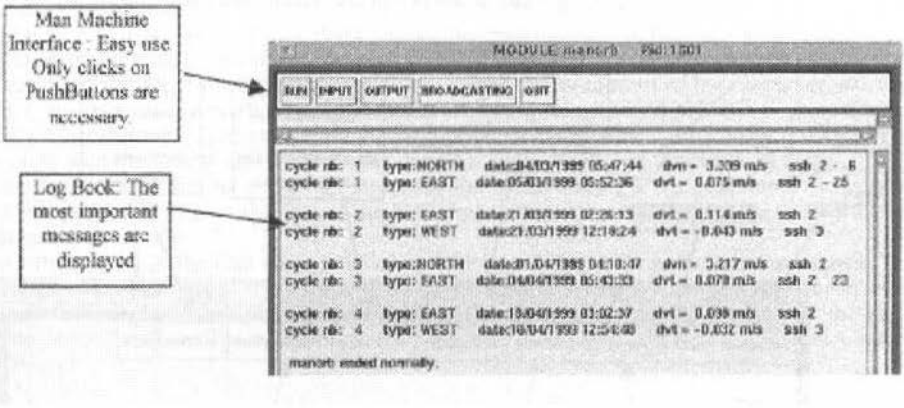


Fig. 5 Man Machine Interface of a Space dynamics module (maneuver calculation)

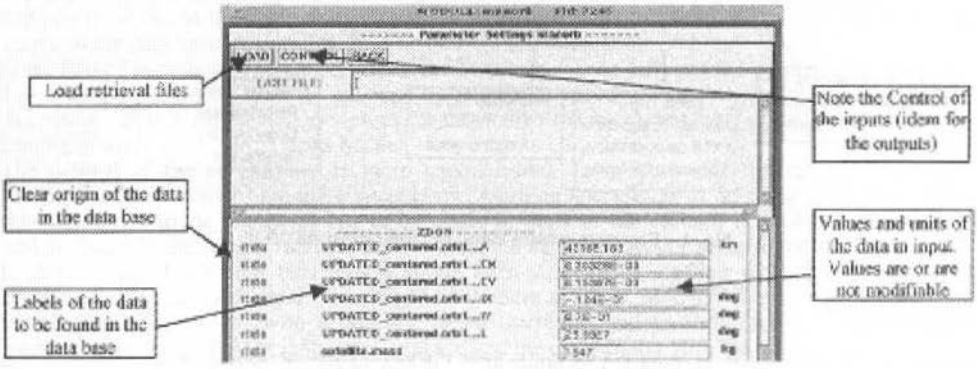


Fig. 6 Parameters Setting of the MANORB module (calculation of the maneuver)

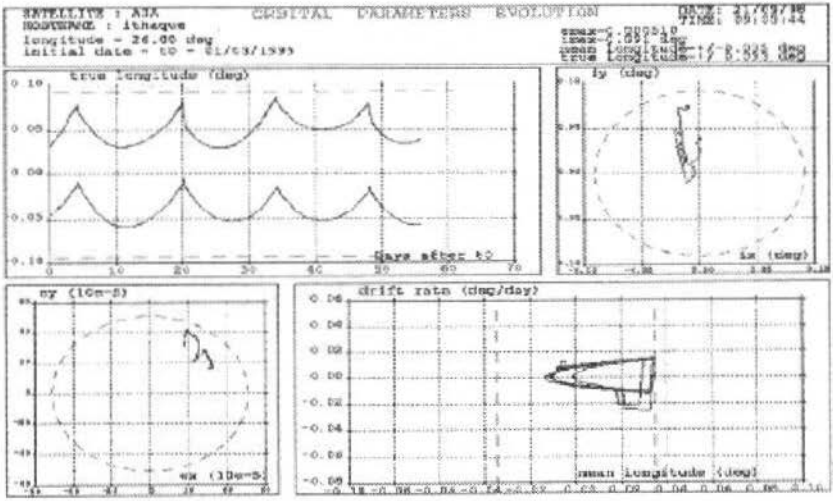


Fig. 7 Example of graphic output: Orbital parameters Evolution over several cycles. A dedicated MMI (XTRACE) is used to modify the graphs (Zoom, Personal touch ...)

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 149

ISSN 0100-7386
Printed in Brazil

Real Time Systems I

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 150-155

ISSN 0100-7386
Printed in Brazil

Test Strategy for a Satellite Control System Developed According to an Object-Oriented Methodology

Ana Maria Ambrosio
Luciana Seda C. Gonçalves
Paulo Eduardo Cardoso
National Institute for Space Research – INPE
Ground System Division – DSS
Postal Box: 515 São José dos Campos -SP 12227-010 Brazil
ana, luciana,paulinho]@dss.inpe.br

Abstract

This paper presents an adaptation of the Hierarchical Approach for Object-Oriented Software Testing, defined by Siegel[1], used in the verification and validation process of the Satellite Control System with the following characteristics: designed according to the Object Modeling Technology (OMT), implemented in C++, providing: a window-based user interface, a Relational Data Base interface, TCP/IP communication protocol interface, and using third party class libraries. This method has being applied in the Telemetry and Telecommand System for the China-Brazil Earth Resource Satellite (CBERS TMTC) developed by the Ground System Division (DSS) at INPE. The experience in applying the test method, the difficulties and the points to be improved are discussed.
Keywords: *Object Oriented System Testing, Satellite Control Center System.*

Introduction

Software systems are more and more complex so to define a strategy and a controlled test process is essential to assure the target system's correctness in an efficient way. When the software is developed according to an object-oriented language extra complexity has to be considered for testing[2]. Hence, it is necessary to introduce and to adapt new techniques in the conventional verification/validation process in order to make building and supporting test process easier.

The DSS team, responsible for the CBERS TMTC system development, has embraced an object-oriented methodology aiming to augment the productivity level and to reduce the cost of the software what led to re-think the verification and validation process. The first orientation in this way was to analyze the testing object-oriented software approach proposed by Siegel[1] which is summarized in following section.

The TMTC main design characteristics and its software components are presented in the next section. The others sections present the strategy and how the TMTC system was effectively tested.

The CBERS TMTC System

The Telemetry and Telecommand System (TMTC) has being developed to provide the requirements of control and monitoring the CBERS satellites family.

This system was designed to perform the following functions:

- process and display the real time and stored telemetry (playback);
- transmit time-tagged and immediate telecommands including pre and post verification;
- restore and display historic telemetry, sent telecommand and events;
- provide facility for configuring the system, through parameters editors, in order to support various satellites without changes in the code.

The TMTC is based on a distributed architecture composed by microcomputers interconnected by a local area network (LAN). This LAN is linked to a private communication network (the RECDAS) by a router which allows the software applications to communicate with the ground station equipment through TCP/IP protocol. In this way, the software applications would be developed so that they would run in Satellite Control Center (SCC) as well as in ground station where they would act as a backup in case of SCC failure.

Windows-NT operating system, C++ language, Microsoft Visual C++ tool, object-oriented methodology and a relational database manager were used in the software development. Standards were adopted as much as possible in order to improve the portability. The API Winsock supported the TCP/IP communication with the Ground Station (GS) equipment; the ODBC (Open Data Base Connectivity)

standard eased the access to the data base system. The Microsoft Foundation Class library (MFC) and a graphic library (Quinn-Curtis) were also used.

The software architecture[3] was organized in five software components as shown in figure 1. Each software component aggregated a set of strongly related information.

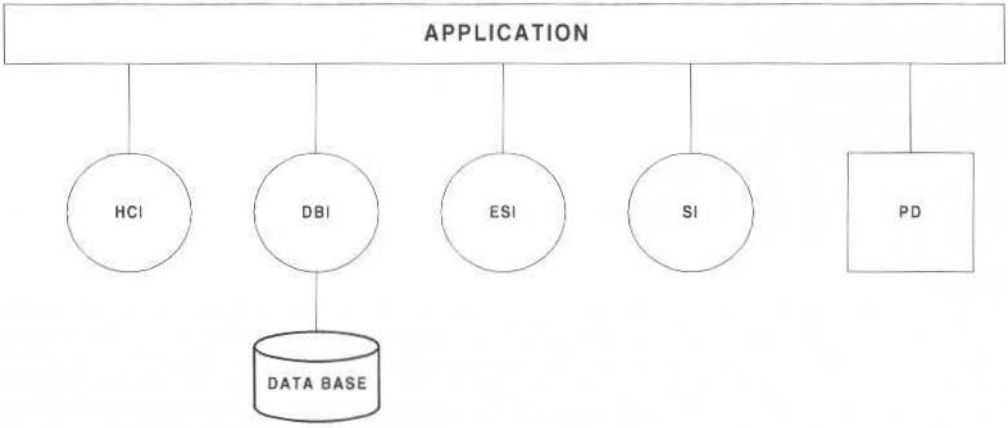


Fig. 1 Software Components

The Data Base Interface (DBI) component is in charge of isolating the data base details from the others components. It represents a shell providing independence of data base managers by using the ODBC standard. This component is composed of 17 interface classes which use 64 *record-set* classes responsible for the access to the data base through the ODBC standard. The *record-set* classes were automatically generated by the Visual C++ ClassWizard tool.

The External System Interface (ESI) component is responsible for the interface between the application and other systems or devices. This component encapsulates the communication protocol, so that any change in the protocol will not affect the others components. Both the TCP/IP communication and the SDID protocol handling services are encapsulated in two basic classes which are specialized in the Problem Domain component to accomplish their specific requirements.

The Support Interface (SI) component provides a friendly interface to the operating system functions and other general services. This component is composed of three ready-for-use classes: time conversion, event log and system resources access and one basic temporization class, which needs to be specialized to comply with the specific requirements of the others components.

The DBI, ESI and SI components may be defined as building blocks representing a set of related and reusable classes designed to provide useful and general-purpose functionality to the system.

The Human Computer Interface (HCI) component is in charge of implementing a friendly interface between the user and the application. This component was divided into two application sets: telemetry with 39 classes e telecommand with 12 classes. These classes were derived from the MFC user interface classes and were generated with the Visual C++ AppWizard and ClassWizard tools. The application allows the users to activate the services of the PD component.

Differently of the others, the Problem Domain (PD) component, directly related to the application domain, had the classes decomposition oriented to functionality. This component doesn't represent a simple collection of independent classes, but rather it is a strongly wired set of interconnected classes. It was divided in 2 sets of classes corresponding to both the telemetry subsystem (32 classes) and the telecommand subsystem (19 classes). These classes provide the necessary services to the satellite monitoring and controlling.

The Data Base was decomposed into two parts. One part, called "*descriptive*", contains all the parameters for configuring the specific problem domain as well as the application as a whole. The other part, called "*historic*", keeps recorded all the received telemetry, the sent telecommand and also the application events.

The SIEGEL Test Approach

In the Hierarchical Approach for Object-Oriented Software Testing, published in Siegel[1], the system is divided into hierarchical levels based on the concepts of software elements: objects, classes, *foundation components*, sub-systems and system. Each element is designated as SAFE whenever it has met its testing

standards specified in the testing plan. Once the element is safe it can be integrated with other safe elements. It is important to observe that "safe" is a relative state and modifications in a safe element will oblige to repeat the tests in that element and in all components where it is used.

The hierarchical approach is focused on the *foundation components* which may be a complete class hierarchy or a cluster of classes performing a core function or representing a logical or physical architectural component. This approach explores the hierarchical nature of the inheritance relationship: any object consists not only of the properties one give it but also of those it inherits from its more general and abstract parents. In testing a hierarchy of classes one reuse the test information of the parent-class to orient the test of the child-classes.

A graphical representation of the hierarchical approach is given in the figure 2. The methods at the base of the pyramid are individually tested before being integrated into classes. The classes are the basic building blocks of the (pyramid) system and must be tested up to success and be safe. Collections of safe classes form the *foundation components*. After testing *foundation components* to a safe level they can be integrated with other safe *foundation components*. In the integration testing of safe *foundation components* only the interconnection of the *foundation components* and the new composite functionality are addressed, eliminating the need of testing all combination of states. The integrated and safe *foundation components* are combined with each other into subsystem and safe subsystems are integrated into the entire system, the top of the pyramid[4].

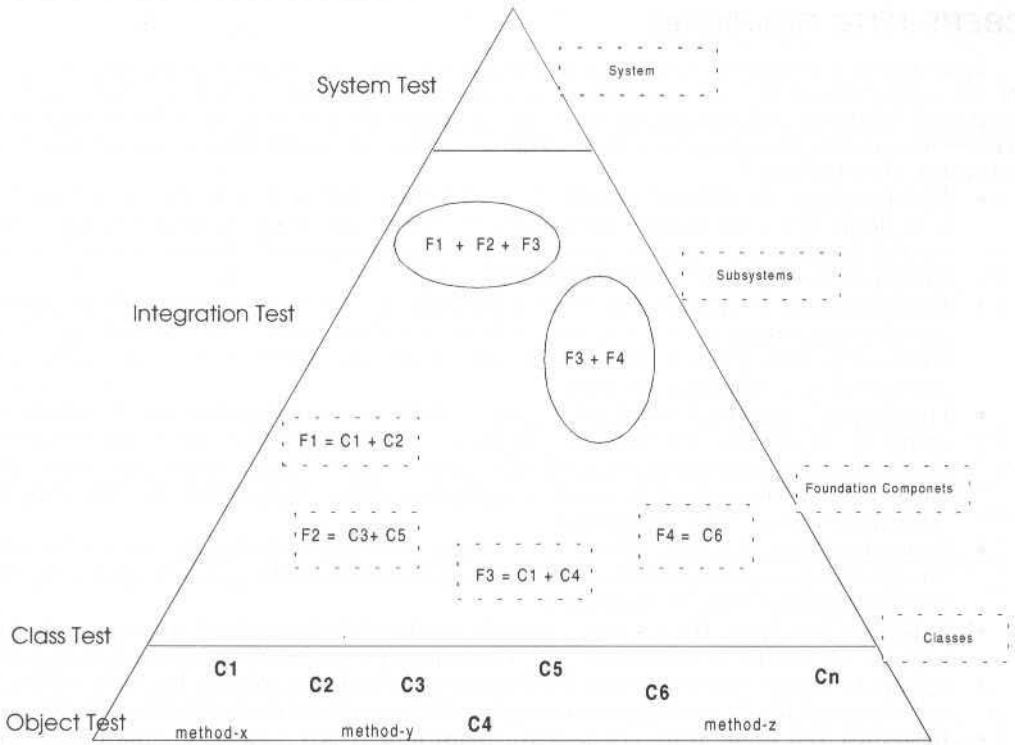


Fig. 2 The hierarchical approach

The hierarchical approach provides a structural set of test suites that make up the structure of the pyramid, as:

- Conditional Test Suite: tests classes using the conditional test model and its accompanying assertions, exceptions, concurrent test operations and message polling tests.
- Hierarchical Incremental Test Suite: tests foundation components using various test models as state-transition, transaction-flow, control-flow, data-flow.
- Integration Test Suite: tests combinations of foundation components.
- System Test Suite: tests systems using system test models.
- Regression Test Suite: re-tests classes, foundation components, sub-systems and system.

The Test Approach To The TMTC CBERS

In order to reduce the effort of designing and applying the tests in the TMTC system some test development rules were established. The first rule was to define a common directory structure to be incorporated in every development microcomputer. The directory has in its first level one sub-directory to each software component and in the next level the folders: *source*, *data*, *doc*, *include*, *library* and *test-project*. Only one microcomputer stores a repository where the safe classes are inserted/deleted exclusively by the configuration controller. However, these classes can be freely ready-only accessed by any developer through the network.

The second rule was to standardize the classes naming and its corresponding files according to the software components and the telemetry and telecommand subsystems as well.

The third rule was related to the test documentation. It was established a minimum set of reports for each test element (class, foundation component, subsystem). The drivers, the stubs, the simulators, the data files and the test cases were reported to each test element. The test cases composing the test suite were documented without the details of how to execute the test.

In the following are presented the test phases and the way the tests of the TMTC system were executed.

CBERS TMTC Test Phases

In this first work related with testing an object-oriented system it was established test phases based on the test suites proposed by Siegel considering the culture and experience of the team. The strategy is incremental bottom-up and only dynamic tests[2] are considered. To each phase should be defined: how generate the test suite, how apply the test (tools, entries, test programs and result logs) and how to get the test verdict. The phases are:

- Unit Test phase: the minimum element to be tested here is the class, however, the methods of a class should be tested in isolation when necessary. The test suite cover the functionality, the class external interface, the exceptions and the concurrency of operations. Test driver and stubs are strongly recommended to be implemented at this phase.
- Class Integration Test phase: in this phase the *foundation components*, composed of safe classes, are chosen according to the scenarios[5] defined during the design. The test suite cover the functionality of the group of classes, the interface among the classes, the states, etc. Test driver and stubs are implemented at this phase.
- Foundation Components Integration Test phase: safe foundation components are integrated with others up to compose the complete subsystem. The test sequence covers the subsystem functionality, the subsystem interface, the subsystem exceptions. Simulators for implementing the interfaces with other subsystems may be implemented and the class drivers and stubs must be substituted by actual classes at this phase.
- System Integration Test phase: all safe subsystems are integrated by testing the system at a whole. The test suite covers the system functionality, the system interface, the system exceptions. System interface simulators may be implemented.
- Validation Test phase: The test suite covers the functional, behavioral and performance system requirements, besides the usability, security and flexibility requirements specified for the system.
- System Test phase: This is the high level test of the system. The software has to be combined with the actual data-base, equipment, and operators to be verified if it satisfies the specification.
- Regression Test phase: regression test, as defined in reference [1], means a series of tests run against a previous version of the element under test (foundation component, system or whatever). The current version exists either because of maintenance (fixing faults in the element) or enhancement (adding features). The regression test had not a special standardization at this first work.

CBERS TMTC Test Execution

The phases are hierarchically built in which the test starts with isolated classes and continues with growing group of classes. Nevertheless, at one time, different test phases can be going on for different group of classes. One strategy was to test the classes in a pre-established order to diminish the number of drivers and stubs to be implemented. The safe classes were used, as much as possible, to test other classes. Other strategy was to use the subsets of test suite of classes to compose the test suite of the corresponding foundation components in later phases.

Particular strategies adopted to each software component are presented below:

a) *DBI, ESI and SI components* - these components were the first to be tested and put safe as they interfaces each other, the PD and the HCI components. In DBI component only the classes that interface the components were tested in isolation, the *record-set* classes were considered safe by having been generated by the own development tool. In ESI and SI components all the six classes were tested in isolation to become safe. In the total 10 drivers were implemented to test every class of these components: 5 for DBI, 2 for ESI and 3 for SI.

b) *PD component* - The tests of this component were divided according to the telemetry and telecommand functions using the safe classes of the components DBI, SI and ESI. During the Unit Test phase, the first classes to be tested were that ones with no relationship to others of the PD component. Next, the classes associated with the classes already *safe* were tested. When the class to be tested had related classes in a not safe state, either the related classes were simulated or the tests were postponed to the Class Integration Test phase.
In the Class Integration phase, the classes were assembled in small foundation components which, after being tested up to become safe, composed bigger foundation components and gradually the foundation component corresponding to the whole Problem Domain was integrated and tested. In the total, 21 drivers, 2 stubs and 2 simulators were implemented to test the classes of the PD component.

c) *HCI component* - The HCI was also divided according to the telemetry and telecommand functions but with a different test strategy. The strategy was to test all the user interface graphical elements (menu, bottoms, dialogs, etc.) and use the safe classes of the other software components, avoiding to simulate them. The HCI testing was oriented to user interface facilities instead of the hierarchical approach.

During the test execution, it was realized that the generation and maintenance of both the class and the *foundation component* drivers are essential for easing the regression tests. When the drivers were lost, by any reason, they had to be re-implemented or an existing driver of a foundation component was used to indirectly test a specific functionality of a class. In the later the test was much more difficult to be applied. In both cases the amount of time spent in the test was increased.

Similarly, the individual test of classes or foundation components can not be neglected. Experience has showed that the use of not safe classes or foundation components caused a great number of problems in the subsequent phases. The later errors were found in the test phases, the more difficult of being detected and diagnosed they were.

Conclusion

This paper has presented a test strategy defined to a system developed according to an object-oriented methodology and successfully applied in the CBERS TMTC system verification and validation. The success of the strategy was confirmed during the Compatibility Test of the Brazilian Ground Station and the CBERS satellite, when, at the first time, the TMTC System was connected to the satellite test model and had all their telemetry and telecommand interfaces verified. In these tests the occurred errors were only related to misunderstanding of the interface specification. No errors in the code were detected showing that the test strategy have pointed out the errors in the early phases avoiding problems in the future.

Although the test strategy has proved to attend the expectation two points in the test aspects should be improved: the documentation and the configuration control.

The decision of reducing and simplifying the test documentation in order to comply with the schedule has done the regression test dependent of the test designer. Additionally, to obey the spiral model of development[6], in which changes and refinements in the classes are constant, and do not adopt a configuration control for supporting the test programs made the test phases more difficult. So to adopt tools for automating the test preparation, the test execution, the regression test, and the configuration control of the test programs are invaluable to reduce the costs and effort of testing an object-oriented system.

The final aim of the DSS team is to established a more general test standard to be applied to any Satellite Control System developed according an object-oriented method, but, implementing a standard requires an ongoing process.

References

[1]Siegel, Shel. Object-Oriented Software Testing – "A Hierarchical Approach". Printed in USA, I Edition – John Wiley

[2]Martins, E., Junho, 1998, "Verificação e Validação" - INF307 - Instituto de Computação - UNICAMP - Campinas - São Paulo.

[3]Gonçalves, L.S.C., Cardoso, P.E. and Ambrosio, A.M., Set., 1998, "Satellite Control Center: a solution for an adaptable system". Proceedings of the International Small Satellite Symposium. Printed in France, Antibes.

[4]Toyota, C. and Martins, E., Junho 1998, "Reutilização em teste de software orientado a Objetos: Metodologia para construção de Classes Autotestáveis", IX conferencia Internacional de Tecnologia de Software: Qualidade de Software -Anais, Curitiba- Paraná- Brasil.

[5]Rumbaugh, J., Blaha, M., Premerlani,W., Eddy, F. and Lorensen,W., 1991, "Object-Oriented Modeling and Design". Printed in USA by Prentice-Hall, Inc.

[6]Gonçalves, L.S.C., Cardoso, P.E. and Ambrosio, A.M., June, 1998, "Lessons Learned in adopting PCs at the Brazilian Satellite Control Center". Proceedings of the SPACEOPS98, Japan.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 156-159

ISSN 0100-7386
Printed in Brazil

JAVA/CORBA Technology Used in Control/Centre Operations: a Component Approach

François Jocteur Monrozier
18, avenue Edouard Belin,
31401 Toulouse Cedex 04, France
Francois.jocteur-monrozier@cnes.fr
Christian Planes, Michel Lopez
18, avenue Edouard Belin,
31401 Toulouse Cedex 04, France

Abstract

The purpose of this paper is to present DIMO: a broadcasting parameters system using CORBA, JAVA and WWW technologies.
The DIMO system is built for dispatching parameters from Operational Centres to Operators during Satellite housekeeping. So, DIMO is a shared multi-mission system needing different configurations for each Satellite housekeeping.
The full paper describes benefits of mixing technologies such as CORBA, JAVA and Internet. We will also discuss about reusable approach that is achieved in this system.
Keywords: *reusability, CORBA, JAVA, WWW, component, broadcasting*

Introduction

The component approach is a way making reusable solutions. The re-use concept is not a new comer in Information Technologies environment. A lot of books and works have related reusability on analysis level (e.g. Design Pattern) and on programming level (with C++, Java, ...). However re-use is difficult to apprehend and set up. Finally re-use seemed to be at too low level (re-use of classes) to measure feedback on its investment. Indeed, obtaining reusable solutions requires some efforts during software analysis and design. Moreover, this effort could be lost by lack of tracability, lack of tools making it possible to the actors to know what is reusable. Does the re-use have real chances to bore? The answer lies in fact into the capability of developing high level reusable components.

Concrete Case

The following paragraphs present a concrete case of re-use realised at CNES for an operational system. This system illustrates the use of relatively new technologies (JAVA, CORBA, WEB) and exposes the advantages of a such solution.

Description of DIMO system

DIMO system is dedicated to **parameter's broadcasting** during Satellites housekeeping mission. DIMO is a **shared multi-mission system** fitting in various configurations for each mission.

The system is flexible and generic, indeed it allows:

- Parameters of **all types** and **all formats** (using IDL Corba),
- Accept any number of producers (centres operational centres such as Flight Dynamic Centre called SDM, Satellite Control Centre called CCS, Orbit Control Centre called COO),
- Accept any number of consumers (operator's stations),
- UI (User Interface) configuration for complex parameters visualisation and for each mission,

The system is composed of three main elements:

- Event handler service of parameters (**GEM** : **G**estion d'**E**vénements inter-**M**achines),
- WWW server allowing the access to the services and information of the system,
- Applet to visualise parameters (**VISIGO** : **VIS**ual**I**sation **G**énérique de paramètres **O**pérationnels)

The following figure gives an overall view of the system:

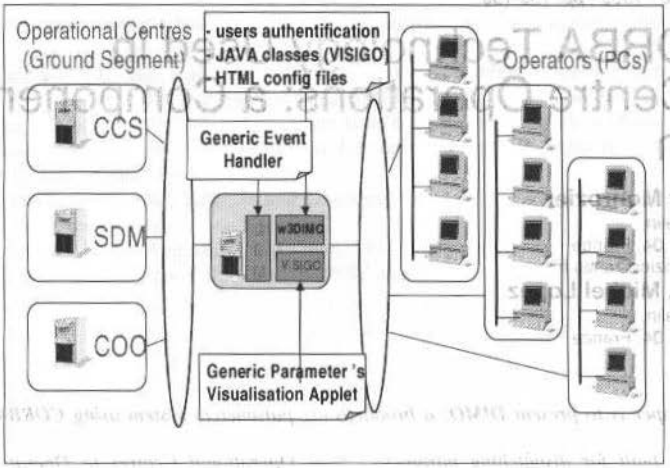


Fig. 1 Overall system view

Event handler Service

An handling event service called GEM (**G**estion d'**E**vénements inter-**M**achines) was specified and prototyped (industrialisation is in progress). This Event handler service is a CORBA server that is based on the Naming and Event service of CORBA. The guiding principles of service GEM are the following:

- publish/subscribe model,
- event filtering (based on the values and the name of the basic cells constituting an event),
- fault-tolerance and transparent reconfiguration for client applications,
- proxy mode to optimise network use (a local GEM service can federate subscriptions of local applications to a remote one).

The following figure presents principles of GEM component:

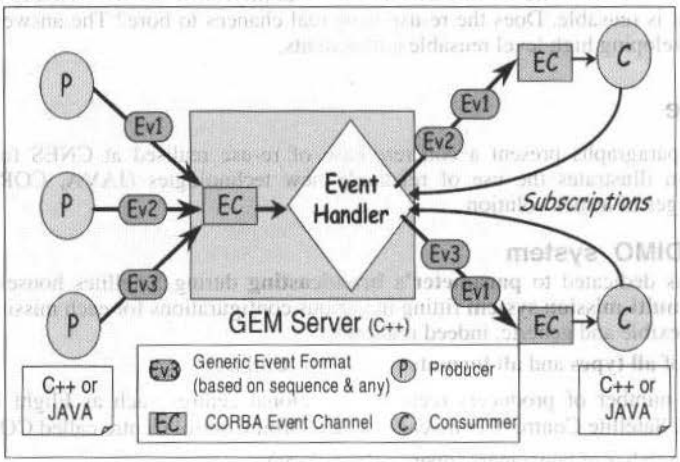


Fig. 2 Event handler principles

Producers send events (parameters, messages...) to event channel of one or more GEM instances. Each consumer (subscribed) has its event channel. While a subscription to a GEM service, consumers give in parameter event channel reference towards which events must be sent, and a filter specifying a "type" of events waited by consumers.

Event formats are similar to a train of basic cells (unlimited number) which can encapsulate any type of data (simple or structured) supported by the " dynamic " types that CORBA offers: **Any and sequence**.

Applet for parameter visualisation

To visualise operational parameters, each operator has a WWW navigator with whom it downloads a dedicated applet called VISIGO that will carry out the subscriptions to one or more event handler services. A simplified version (prototype) has been developed by CNES. The applet is configurable to allow several types of visualisation. The following image shows a table view:

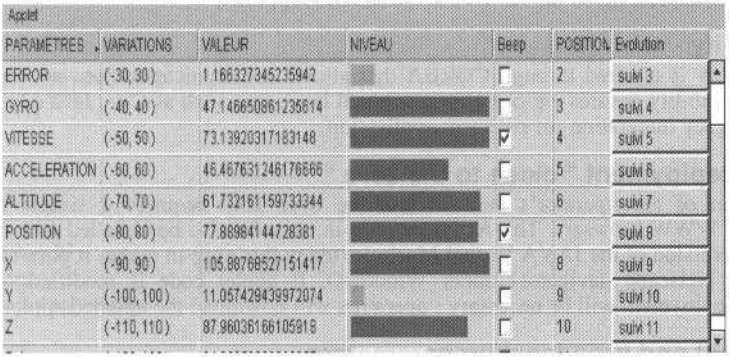


Fig. 3 A table view exemple

To achieve this, we defined three elementary graphic components:

- curves,
- tables,
- messages panel (simple window text).

These elementary graphic components will be implemented in the form of JavaBeans. These elementary components are arranged in display windows. Operators can compose their synoptic through display windows.

Each component has an HTML file configuration allowing a modular and an increased simplicity of the use of the windows and elementary graphic components. The following figure illustrates principles of applet VISIGO:

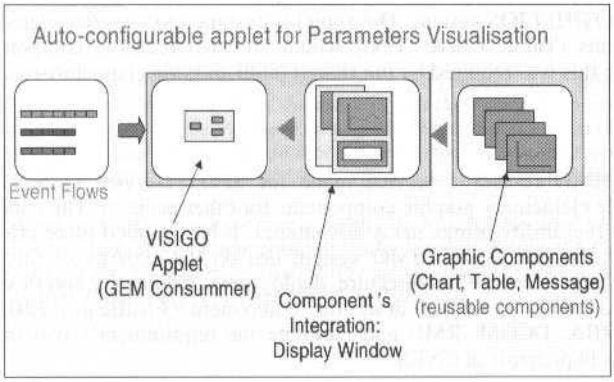


Fig. 4 Visigo applet principles

HTML file/tags are used to configure the following elements:

- subscriptions to one or more event handler,
- formats of the events to be received,
- parameter's description (name, gauge, alarm, extraction of parameter value from event),
- number of display windows,
- configuration of each elementary graphic component and display window.

Accessing to DIMO services is done through a WWW server allowing operator's authentification (login, password, IP address).

Advantages of Dimo System

Portability

Consumer and producer applications are written in JAVA. So, various operating systems (NT and Unix) can be used depending on the Java Virtual Machine support.

Interoperability

Interoperability is ensured through CORBA that allows communication between Unix applications and NT, interoperability ensures various object-oriented languages such as C++, Java can be used. This is a key point of legacy integration for producer application.

Cost of the deployment " close to the zero "

Configuration of the operator's stations (can be over 200) requires a simple installation and configuration of WWW browsers. The Activator plug-in must be also be installed making it possible to support a newer version of the JAVA Virtual Machine (today this plug-in makes it possible to support the JDK1.2 for the Netscape navigators under Windows). If applet code is modified (evolution, bugs correction), no deployment will be necessary : operators will download the updated applet at the next use.

Generic format and broadcasting parameters

Each operational centre must broadcast specific formats of data: the system is able to take into account any format type of data without changing code of applications. Formats and types of the conveyed data form part of configuration parameters of producer and consumerapplications.

Overload and robustness

Software architecture can easily evolve/move according to the volume of data to convey, to the number of producers and to the number of consumers. It is possible in DIMO system to install several machines ensuring the broadcast service for several and simultaneous missions. Moreover, GEM service is designed to support fault tolerance. Thus a redundant service can be defined for each (nominal) service.

The service swing is transparent for client applications.

Reusable components

Service GEM is a reusable component. Indeed, it also will be used for the messages broadcasting (log message types) in SPOT/HELIOS system. The applet is designed to integrate elementary

graphic components (Table, Curve, Text) which are also reusable components. Components are strongly configurable: this was required by the shared multi-mission characteristic.

Conclusion

DIMO System DIMO re-uses a service made for SPOT (service GEM) and must lead to the availability of reusable elementary graphic components for other projects. The concrete case illustrated in this paper shows that reusability brings many advantages. It has implied more effort during analysis and design phase. Indeed, the interest of DIMO system lies on the synergy of three technologies, which induces many advantages in terms of architecture, deployment, modularity and flexibility.

" New " standards unify means of designing component (JavaBeans, EJB, ActiveX), means of communication (CORBA, DCOM, RMI) and facilitate the requirement's widening. The evaluation of these new standards is in progress at CNES.

RBCM - J. of the Braz. Soc. Mechanical Sciences ISSN 0100-7386
Vol. XXI – Special Issue - 1999 - pp. 160-167 Printed in Brazil

User-Centered Design of Spacecraft Ground Data Systems at Nasa's Goddard Space Flight Center

Jeffrey Fox
Pacific Northwest National Laboratory[1]
901 D Street SW, Suite 900
Washington, DC 20024
jeff.fox@pnl.gov

Julie Breed
Karen Moe
Robin Pfister
Walter Truszkowski
Dana L. Uehling
NASA/Goddard Space Flight Center
Greenbelt Road
Greenbelt, MD 20771
julie.breed@gsfc.nasa.gov
karen.moe@gsfc.nasa.gov
pfister@killians.gsfc.nasa.gov
truszkow@pop500.gsfc.nasa.gov
dana.uehling@gsfc.nasa.gov

Adriane Donkers
User Works, Inc.
1738 Elton Road, Suite 138
Silver Spring, Maryland 20903
adonkers@userworks.com

Elizabeth Murphy
Bureau of the Census
Statistical Research Division
Washington, DC 20233
elizabeth.d.murphy@ccmail.census.gov

Abstract

Spacecraft ground control data systems are by necessity sophisticated and complex. One of the challenges for system designers is to understand and incorporate the needs of the users so that their systems are easy to use and minimize user errors. User-centered design (UCD) can help achieve these goals. UCD techniques can be applied to the design of ground systems to improve quality and functionality so that the users can perform their work more effectively and efficiently. This paper provides a high-level survey of UCD efforts at NASA's Goddard Space Flight Center (GSFC).

The paper describes some of the techniques that have been used and the benefits they have produced via case studies. The goal of the paper is to share lessons learned from performing user-centered design at NASA/GSFC with the wider ground systems community and to provide pointers for interested readers to find more detailed information.

Keywords: *User-Centered Design, Prototyping, Cognitive Modeling, Usability.*

Introduction

The ultimate impact of a system, no matter how innovative its design or capabilities, depends on how well its users interact with it. Often, as systems become more complex, they become harder to use. An example of this is mission operations software that requires Spacecraft Control Team (SCT) members to monitor thousands of parameters distributed across multiple screens.

Modern ground control software systems employ graphical user interfaces, expert systems, software agents, and data visualization in an effort to alleviate these problems. However, each of these approaches also introduces its own new demands on the users. In addition, new operational models that reduce the number of staff or even eliminate full-time operations (i.e., "lights-out" operations) provide new challenges and can significantly increase the workload for the remaining limited on-call staff.

---

[1] Pacific Northwest National Laboratory is operated for the U.S. Department of Energy by Battelle under Contract DE-AC06-76RLO 1830.(GSFC).

In his seminal book *Designing the User Interface: Strategies for Effective Human-Computer Interaction*, Shneiderman[1] notes that the degree to which a system meets the needs of its users can be measured via attributes such as:

- Time to learn
- Speed of performance
- Rate of errors by users
- Retention over time
- Subjective satisfaction

To optimize these attributes of a system, the design of the system, as well as the process of designing the system,[2] must focus on the users and their needs. This is known as user-centered design (UCD).

The goal of UCD is to ensure that systems are designed with consideration of their users' capabilities and limitations. UCD considers the software, hardware, users' tasks, environment, and all their interactions. Properly employing UCD will increase the usability of those systems, making them easier to use, increasing user satisfaction, and reducing error rates. UCD concepts come from many fields, but most commonly from human factors engineering. In addition, UCD has an ever-increasing impact on overall systems design as the percentage of the software code that is dedicated to the user-system interface increases.

Various UCD techniques and activities are used at NASA/GSFC to design high quality mission operations software and advanced control center environments. Table 1 lists a sampling of some of those UCD techniques. Table 2 lists some projects that have benefited from the application of those techniques.

Which techniques are used and when they are used depends on the type and size of the project (e.g., style guides are most useful for large development teams to control consistency, while expert evaluations are equally effective for any size project). Likewise, the

Table 1 UCD Techniques Used at NASA/GSFC

• User design working (focus) groups
• Cognitive modeling
• Education of software designers
• Expert evaluations
• Rapid software prototyping
• Scenario-based design
• Task analysis
• Usability testing
• User interface guidelines, style guides and standards
• Workstation and control room design

Table 2 Projects employing UCD

• EOSDIS
• Data Distribution Facility
• Desktop Satellite Data Processor
• Hubble Space Telescope
• Spacecraft Emergency Response System (SERS) for SMEX and MIDEX missions

positive effects of some techniques are maximized when they are used at certain phases in a project's lifecycle (e.g., task analyses are most useful when performed before any new software or hardware is developed). Similarly, the type of project (R&D vs. deployment) influences technique selection.

The remainder of this paper describes several of the UCD techniques that have been successfully used at NASA/GSFC in the design of software. Unfortunately, there is only space to discuss a few of these techniques: cognitive modeling, software rapid prototyping, and usability testing. Along with the descriptions are examples of how these techniques were used on actual projects and the positive results of using these techniques. The references cited provide opportunities for further learning about UCD.

As you read through these different techniques, you will note common themes and goals. You should also note that these techniques are complementary and, when used together, produce an even more usable design.

Cognitive Modeling

An important goal of user-centered design is to understand the users and how they interact with the systems that they use. Using both syntactic knowledge (knowledge of procedures, such as pressing the "delete" key to erase a character) and semantic knowledge (knowledge of a domain, such as a theory) gained by experience with a system, users build a mental model of how that system operates. McGraw[3] defines a mental model as an organized set of domain concepts, their relationships and functionality. In computer-based systems, a user's model of how to perform a task is directly influenced by how she perceives those data being presented by the computer. For example, let's say that in order to successfully accomplish a task, a particular switch on a spacecraft must be open. The spacecraft controller must not only know that the switch is to be open, but also be able to interpret the current state of the switch as displayed on the computer's screen (perhaps displayed graphically as a circuit diagram or simply as a "yes/no" field). Also she must know how to tell the computer to send the command to open that switch (perhaps via a typed command or by clicking on a button).

The task flows more smoothly when the data displayed and the method of interaction match the users' expectations of how the system will operate, that is, their mental models. Thus, it is important that the designer incorporate the user's model of the task into the design of the computer's interface so that there is no disconnect between what the operator thinks she sees and does, and what the computer actually is presenting and executing. This problem is compounded by the fact that, to a certain degree, all users possess somewhat different mental models.

In contrast with mental models, a cognitive model is a researcher's attempt to represent conscious and subconscious mental processes and outcomes. A cognitive model is a theory-based, graphical representation of inferred relationships between hypothesized components of human thought. These overt models of the mind may be qualitative or quantitative, specific or general, and they are typically embedded in graphical models of human performance (e.g., Wickens[4]). Most cognitive models encompass the basic constructs of human information processing: sensory processing, perception, short- and long-term memory, and analytic processes such as problem solving and decision making. Different models vary in their level of detail and, sometimes, in the theory that underlies their depiction of the flow of information and control within the model. For detailed treatments of cognitive models in the psychological and human factors literature, see Best,[5] Eberts,[6] Newell,[7] and Solso.[8]

Cognitive models aid analysts in getting beyond the behavioral aspects of human activities to their underlying goals, objectives, and thought processes. Cognitive modeling provides a theory-based framework for cognitive task analysis; which documents operational decision points, information requirements, and the analytic processes followed in making decisions on the basis of available information, incomplete and uncertain though that information may be. Cognitive task analysis provides critical input to user-interface design and evaluation. Such models serve as frameworks for investigating and documenting the role of human information processing in operational settings. They help in the process of understanding the problem-solving and decision-making activities performed in current spacecraft ground control environments. Sources on cognitive task analysis include Redding[9]; Roth and Woods,[10] Roth, Woods, and Pople,[11] and Schlager, Means, and Roth.[12]

In a specific operational environment, early and continuing cognitive modeling and cognitive task analysis can help to identify analysts' information requirements and can support the design of automated job aids. When a job or position is being defined or redefined, it is worth the effort to document cognitive tasks and information requirements as a basis for the design of user interfaces, position manuals, and training programs. When a change is contemplated for a computer system, design of user interfaces and operational procedures that are based on a thorough cognitive task analysis is likely to yield a more usable and operationally suitable product than will result from design that gives low priority to human information-processing issues. A design's support for cognitive capabilities and compensation for cognitive limitations should be evaluated throughout the project's lifecycle.

Example

At NASA/GSFC, cognitive modeling has provided a basis for developing an automated approach to predicting the demands imposed on operators' cognitive resources by various aspects of user interfaces. Modeling the cognitive demands imposed by user interfaces led to development of a series of Computer-Human Interaction Models (CHIMES) prototypes. Early CHIMES prototypes demonstrated the feasibility of assessing the cognitive demands imposed by text-based user interfaces. The underlying model included a functional hierarchy of operational activities, from subtask to mission, and a set of cognitive attributes associated with each level of the functional hierarchy.[13] A demand analysis using CHIMES could be performed early and iteratively throughout the design and implementation of a textual user interface. CHIMES evolved into a tool for checking a user-interface design both for internal consistency and for compatibility with design rules.[14]

The early CHIMES methodology and toolset were applied in an analysis of the user interface to the Planning and Resource Reasoning (PARR) tool, a scheduling aid developed by the Bendix Field Engineering Corporation.[15] The purpose of the analysis was to evaluate the PARR user interface from a human factors perspective in order to improve its ability to support efficient, effective scheduling of spacecraft events by the Earth Radiation Budget Satellite (ERBS) scheduler. Analysts observed and interviewed the ERBS scheduling operator about her functions and tasks and examined the various paper-based job aids that she consulted in building the weekly schedule. The scheduler's job was modeled as it was currently being done, in a largely manual mode, and as it could be done with the aid of the PARR tool. The analysis permitted a comparison between the largely manual job and the aided job, showing that task demands on the ERBS scheduler were significantly lower after the introduction of the automated scheduling aid. The analysis identified several PARR displays that could benefit from some redesign.

In the current spacecraft ground control environment, with its emphasis on increased automation to reduce mission costs, cognitive modeling and cognitive task analysis can provide inputs to decisions on system design. It cannot be assumed that partially or fully automating a function will eliminate the need for human involvement in mission operations. For the success of lights-out automation, it is essential to define the cognitive tasks that operators and analysts currently perform, many of which are not fully documented in existing procedural manuals. If fault detection and resolution are to be automated, cognitive models of humans performing those tasks can provide valuable guidance. Cognitive issues in lights-out automation are under investigation from several conceptual and empirical perspectives (e.g., Mitchell, Thurman, & Brann,[16] Murphy & Norman,[17] Truszkowski[18]). These investigations are likely to lead to the development of cognitive models of the combined cognitive system, that is, the cooperative activity of intelligent software agents and human analysts, where mutual understanding will be critical to success.

Software Prototyping

Traditional software development is a very formal and structured process. This process is generally represented by the "waterfall" model that starts with requirements analysis and is followed sequentially by design, development, testing, operations, and finally maintenance. For this approach to be successful, all of the detailed requirements must be known and documented prior to the onset of design. Any missed requirements or functionality tend not to be discovered until late in the product's life cycle during testing or even after deployment. Changes at these points tend to be very time-consuming and costly.

Software rapid prototyping can help alleviate this problem. A rapid prototype is used to simulate a system's functionality and user interface and can serve as a model for the technical demonstration of a system. UCD is specifically concerned about understanding the needs of the end user in order to get a better understanding of requirements early on in the project. The most common reasons for software prototyping are to gain a better understanding of the users' requirements and to allow the developer to confirm that a specific approach will accomplish the needed functions with adequate system performance. Generally, prototyping is iterative and leads to a more cyclic design approach.

Prototyping generally takes one of two forms: "throwaway" or "evolving."[19] A throwaway prototype is usually discarded once it has served its purpose. For evolving prototypes, the functionality is added and improvements are made until the prototype becomes the operational system.

Alternatively, Pressman[20] classifies prototypes into three types - a model, a working prototype, and an existing program. The model, either paper- or computer-based, is a mock-up that depicts the user interface in order to convey a sense of look, feel and functionality of the system. The working prototype is an implementation of specific functions of the system. In the case that a system is already in place and meets all or most of the needed functionality, it can serve as the basis for prototyping additional functions or improving on existing functions.

According to Pressman, an important factor in rapid prototyping is working with the user to define and agree on the scope, purpose, and nature of the prototype - particularly whether it is to be discarded or turned into an operational system.

Boar[21] describes factors, such as the characteristics of the project and the user as well as the application area and the complexity, that can be assessed to determine whether a software project can benefit from prototyping. Once it is decided that a prototype is needed, the form of prototype should be considered.

Example:

The overall purposes of prototyping in the Earth Observing System Data and Information System (EOSDIS) are:

- To address risks to development
- To enable technology transfer into the system and out to the global change research community

- To reduce long term costs through evolution.

The specific goals are to better understand the needs of the user community and to mitigate identified risks. For EOSDIS, multiple approaches are used. Most prototypes investigate system functionality that will later be infused into the EOSDIS Core System (ECS). All of the prototypes start out as intended throwaways, but some that show promise for infusion are evolved to the point that they can be integrated into the system.

One prototype, known as EOSDIS Version 0, began in the early requirements definition phase of ECS to show proof-of-concept for interoperability among distributed systems and to better define interoperability requirements to be levied on the ECS. After a successful demonstration in 1992, this prototype evolved in nature and purpose as functionality was added. It started as a prototype to define requirements and later evolved into a prototype that served as a demonstration of technical capabilities, and then to an operational system. Every six months during its evolution, an on-line, pseudo-operational version of a complete end-to-end system was evaluated by user community representatives. These representatives stated their likes and dislikes, gave recommendations on future improvements, and as a group they agreed on the priorities of functions to be added in the next phase.

While Version 0 evolved, the ECS contract was defined and an end-item deliverable contract was awarded. The development was to follow a formal systems engineering lifecycle process. However a few components that involved end-user interfaces were put on an incremental track with the intention of getting early user feedback on "uncertain driving requirements." The Incremental Track Design process involved various mechanisms to elicit user feedback on prototypes. The key mechanisms included Evaluation Packages, Prototype Workshops, a Client Design Working Group, and Client Workshops.

Evaluation Packages (EP) were a delivery and evaluation mechanism for incremental and other prototype developments. The goal was to maximize visibility for end users, data archive staff, and NASA to get feedback and acceptance. The key challenge of this process was to provide just the necessary amount of structure to enable an evaluation without creating an administration overload.

Prototype Workshops (PW) were typically held after each EP so that each PW included one EP plus other selected prototypes that appeared particularly promising. The PW was held to allow "tirekickers," data archive representatives, and NASA to review all of these latest tools. Surveys were conducted during the PW to evaluate each of the presented prototypes.

Because there were so many functions for which the designers needed more insight, and the fact that EPs and PWs took time and resources to pull together, a Client Design Working Group (CDWG) was formed to generate scenarios, workflows and models (paper and computer mock-ups) for the remaining client functionality.

Some prototyping efforts were more successful than others. The Version 0 system was very successful and is still operational today. The CDWG also was very successful and ensured buy-in of the end-user community. Incremental Track prototyping succeeded in identifying broad issues in the overall system architecture. It highlighted difficulties in installing and maintaining distributed systems. It also helped the project identify the threshold of the tolerance of the end user's burden to gain access to the system. Currently, the Version 0 system is being augmented to support ECS functionality and will serve as the end-user interface for EOSDIS.

Usability Testing

Usability testing is one method of evaluating how well a system meets its users' needs. Usability tests can be performed at any stage of a product's lifecycle, from the paper prototype stage to operations. The main characteristic of a usability test is that the proposed user of the system is observed while performing a series of tasks with a fully functional system or with either a software or paper prototype. It is not a demo to the user, but a chance for the user to work with the system "hands-on."

The tasks usually include those that will be frequently performed or that are most critical. While the user performs these tasks, many types of data are collected. Some of the quantitative measures include:

- Time to complete a task
- Number of errors
- Number of times help is required
- Time to locate specific information

Some of the qualitative measures include:

- Observed frustration
- Participants' comments
- Ratings
- Severity of error

These data are then analyzed to determine where the system is making it difficult for the user to complete the tasks. Suggestions for changes to the system are then presented to the project development team. Alternatively, developers can act as observers of the actual tests to get more direct user feedback and gain insight into the usability testing process.

Usability testing is one component of usability engineering,[22] which takes a full life cycle approach to managing the user interface. Usability testing can take many forms. Ideally, usability testing should occur at regular intervals or at major milestones in a project's life cycle, especially for large, complex systems. This is known as formative usability testing.[2] Alternatively, a system may only be tested as part of a final deliverable. This is what happens for many systems due to funding constraints or the late inclusion of a usability engineer on the project team. This is known as summative testing.[2] The disadvantage of this latter approach is that the results of the testing often arrive too late to be incorporated into a revised design.

Usability testing can be done in a formal lab, with a one-way mirror, video recorders, audio recorders, screen-capture software, and an intercom system; however, a "discount" [22-24] approach to usability testing typically is used at NASA/GSFC. In the discount approach, the tests are often run in the users' actual offices where they complete tasks on their own computers or with paper prototypes. The observer(s) sits beside the user and records data either on paper or on another computer. Even with this "low tech" approach, designers gain valuable information that allows them to greatly improve their applications.

Example

Usability testing for the Hubble Space Telescope (HST) exemplifies a discount approach. In this project, the control center software is currently being modernized. Throughout the development cycle of the windowing graphical user interface, various usability tests were conducted. In one of the first tests, three users sat at a computer and completed a series of tasks while the observer took notes. The tasks included (1) creating a new real-time page (real-time pages are graphical pages that can contain hundreds of updating data points, referred to as mnemonics, that are used to monitor the HST's health and safety), (2) adding components (e.g., alphanumeric fields, stripcharts, labels) to a page, (3) modifying characteristics of the components, and (4) saving the page. After observing the users, the observers identified several problems that needed to be addressed. The problems were not critical, but the design could be greatly improved if they were fixed. Examples of some of the problems included:

- Inconsistencies (e.g., changing the color of mnemonics was done differently in two different dialogs)
- Lack of prompts to guide the user (e.g., no confirmation message when logging in, no indication of how to enter a search string)
- Lack of use of standardized user interface conventions (e.g., use of gray to indicate those menu items not currently availability)
- •Dispersion of related functions (e.g., information that was related and worked together was separated into two windows)
- Grouping of unrelated functions (e.g., editing of two components was done in the same window and it was thought the options worked together when they were, in fact, independent)
- Missing features (e.g., ability to modify several components at one time, ability to zoom to see greater detail).

The feedback that was gathered from the users and the suggested design modifications were reviewed with the developers. In some cases, the suggested changes were accepted and could be easily implemented. In other cases, further information from the users was requested. Some concepts were modeled using a graphics package, and together with the developers the design was modified. Many of the problems in this initial study have been addressed; and, in follow-up meetings, users reported that the design was much improved.

Another example illustrates usability evaluations conducted with paper prototypes. Here the users were shown a paper mock-up of the tool and were asked to verbally describe how they would use it. This technique of capturing the user's thoughts is called protocol analysis.[25] One study evaluated a tool intended to warn flight operation controllers of anomalous conditions via the display of mnemonics. The results showed that users could easily understand the tool. However, it was found that a particular action of the tool was not anticipated and did not match the user's mental model of the system.

More specifically, the tool represents an anomalous condition by displaying a red light for the sub-system that contains the out-of-limits mnemonic(s). The system then allows the user to click on the light to display a second level indicator that shows which sub-sub-system contains the mnemonic(s). The users liked the concept of seeing the warning light and then clicking on it to get additional information pertaining to the anomaly. However, they had other ideas for what additional information should be presented.

Instead of seeing the sub-sub-system, the users preferred to get further detail on the mnemonic (e.g., current value, and highest and lowest values). Once they had more detail on the mnemonic they would then like to have additional options for further analysis, such as the ability to plot the most recent data or examine the limits of the mnemonic. This simple evaluation with a paper mock-up only required about 15 minutes of each of the operators' (5 total) very valuable time. Yet, in this short period of time, a problem with a critical component of the software was identified and a solution recommended. Early identification saved the developer considerable time in code development and thus provided a large return on investment.

Conclusion

This paper has provided just a glimpse at user-centered design and its positive effects on the design of ground data systems at NASA/GSFC. You are encouraged to see the large set of papers and books that provide more details on the UCD techniques referenced in this paper, as well as books on other techniques listed in the introduction[26-29] and some more general UCD references.[30, 31] Also, you may wish to visit the Web sites of some of the larger professional societies focussing on UCD:

- The Human Factors and Ergonomics Society (HFES), http://hfes.org/
- The Association for Computing Machinery Special Interest Group on Computer-Human Interaction (ACM SIGCHI), http://www.acm.org/sigchi/.
- The Usability Professionals' Association (UPA), http://www.UPAssoc.org/.

You are invited to contact the authors via email to learn more about user-centered design or to discuss the material presented in this paper.

References

[1]Shneiderman, B., 1986, "Designing the User Interface: Strategies for Effective Human-Computer Interaction". Reading, MA: Addison-Wesley.

[2]Hix, D. and Hartson, H. R., 1993, "Developing User Interfaces: Ensuring Usability Through Product & Process". New York: Wiley.

[3]McGraw, K. L., 1993 "Designing and Evaluating User Interfaces for Knowledge-Based Systems". New York: Ellis Horwood.

[4]Wickens, C. D., 1992 "Engineering Psychology and Human Performance (2nd ed.)". New York: HarperCollins.

[5]Best, J. B., 1992 "Cognitive Psychology (3rd ed.)". New York: West Publishing.

[6]Eberts, R., 1997 "Cognitive modeling. In G. Salvendy (Ed.), Handbook of Human Factors and Ergonomics (2nd ed., pp. 1328-1374)". New York: Wiley.

[7]Newell, A., 1990 "Unified Theories of Cognition". Cambridge, MA: Harvard University Press.

[8]Solso, R. L., 1991 "Cognitive Psychology (3rd ed.)". Boston: Allyn and Bacon.

[9]Redding, R. E., 1989, "Perspectives on cognitive task analysis: The state of the art. Proceedings of the Human Factors Society 33rd Annual Meeting (pp. 1348-1352)". Santa Monica, CA: Human Factors Society.

[10]Roth, E. M. and Woods, D. D., 1990, "Analyzing the cognitive demands of problem-solving environments: An approach to cognitive task analysis". Proceedings of the Human Factors Society 34th Annual Meeting (pp. 1314-1317). Santa Monica, CA: Human Factors Society.

[11]Roth, E. M.; Woods, D. D.; and Pople, H. E. Jr., 1992, "Cognitive simulation as a tool for cognitive task analysis". Ergonomics, 35, pp. 1163-1198.

[12]Schlager, M. S.; Means, B.; and Roth, C., 1990, "Cognitive task analysis for the real(-time) world". Proceedings of the Human Factors Society 34th Annual Meeting (pp. 1309-1313). Santa Monica, CA: Human Factors Society.

[13]Sheppard, S. B. and Murphy, E. D., 1988 "CHIMES: Computer-Human Interaction Models (Report prepared for NASA-Goddard under Contract No. 018750, Task 29-116)". McLean, VA: CTA INCORPORATED.

[14]Jiang, J.; Murphy, E. D.; Carter, L. E.; Bailin, S. C.; and Truszkowski, W., 1994, "Automated evaluation of graphical user interfaces (GUIs) using CHIMES. Proceedings of the Second Annual Mid-Atlantic Human Factors Conference (pp. 57-63)". Washington, DC: George Mason University/NASA-Langley.

[15]Murphy, E. D., 1989, "Application of CHIMES to the Planning and Resource Reasoning (PARR) Tool (Report prepared for Computer Sciences Corporation under Contract No. 018750, Task 29-116)". McLean, VA: CTA INCORPORATED.

[16]Mitchell, C. M.; Thurman, D. A.; and Brann, D. M., 1998 "The human factor in 'lights-out' automation: Using field study data to identify critical human factors design issues". Proceedings of the Human Factors and Ergonomics Society 42nd Annual Meeting (pp. 364-368). Santa Monica, CA: Human Factors and Ergonomics Society.

[17]Murphy, E. D. and Norman, K. L., 1998, "Beyond supervisory control: Human performance in the age of autonomy". Third Automation Technology and Human Performance Conference Abstracts (p. 26). Norfolk, VA:

Old Dominion University; full paper to appear in M. Scerbo (Ed.), Automation Technology and Human Performance. Hillsdale, NJ: Erlbaum.

[18]Truszkowski, W. 1996, "Lights out" operations: Human/computer interfaces/interactions (HCI). 1996 Technology workshop – Autonomous "lights out" operations workshop: Operational challenges and promising technologies (Presentation viewgraphs, pp. 355-367). Greenbelt, MD: Mission Operations and Data Systems Directorate, NASA-Goddard Space Flight Center.

[19]Smith, M. F., 1991, "Software Prototyping: Adoption, Practice, and Management". New York: McGraw-Hill.

[20]Pressman, R. S., 1992, "Software Engineering: A Practitioner's Approach (3rd edition)". New York: McGraw-Hill, 1992.

[21]Boar, B. Application Prototyping: A Requirements Definition Strategy for the '80s. New York: Wiley-Interscience, 1984.

[22]Nielsen, J. Usability Engineering. San Diego CA: Academic Press, 1994.

[23]Szczur, Martha, Usability testing on a budget: A NASA usability test case study, Behaviour & Information Technology, Vol. 13, No. 1, 1994.

[24]Uehling, D., Usability Testing Handbook. NASA DSTL-94-002, 1994.

[25]Ericsson, K. A. and Simon, H. A. Protocol Analysis. Cambridge, MA: The MIT Press, 1984.

[26]Carroll, J. M. Introduction: The scenario perspective on system development. In J. M. Carroll (Ed.) Scenario-Based Design: Envisioning Work and Technology in System Development. New York : Wiley, 1995.

[27]Gilmore, W. E.; Gertman, D. L.; and Blackman, H. S. User-Computer Interface in Process Control: A Human Factors Engineering Handbook. Boston: Academic Press, 1989.

[28]Kirwan, B. A & Ainsworth, L. K. (Editors). Guide to Task Analysis. London: Taylor & Francis, 1992.

[29]Mayhew, D. J. Principles and Guidelines in Software User Interface Design. Englewood Cliffs, NJ: Prentice Hall, 1992.

[30]Salvendy, G. (Editor). Handbook of Human Factors 2nd Edition. New York: Wiley, 1997.

[31]Sanders, M. S. and McCormick, E. J. Human Factors in Engineering and Design. New York: McGraw-Hill, 1992.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 168-174

ISSN 0100-7386
Printed in Brazil

Standard Access to Space Ground Segment Data Processing Applications : a JAVA Experiment for SPOT4.

Mourad Ould
Centre National d'Etudes Spatiales (CNES)
18, avenue Edouard Belin, 31401 Toulouse Cedex 4 France
ould@cnes.fr

Abstract

The interest to offer a standard access to a software, i.e. the same run procedure one any kind of hardware platform (PC, WorkStation...), exists for much data processing applications in a space ground segment. It is the case, in particular, for client/server ones or for all those comprising an important MMI part such as data visualization applications (telemetry monitoring, trending analysis,...). This standard access constitutes with no doubt an effective means to reduce the exploitation and development costs since it makes it possible to absorb with less expenses the hardware configurations and basic software evolutions.
But how to offer a standard access to an already existing data processing application ?
The purpose of this paper is to answer this question, through a concrete and conclusive experience. It presents the results and brings back the experience of the redevelopment in Java of the activity visualization function of the SPOT4 Earth Observation Satellite's Control Center.
Keywords *: Standard access, Object Oriented Technology, Java, Refurbishing, Portability, Cost reduction.*

Introduction

The experimented data processing application : "Agenda"

The experimented data processing application related to this experience, AGENDA, is an automatic work scheduler. Through an advanced, ergonomic and efficient MMI (X11/MOTIF based), AGENDA offers all the required functions to carry out the operational activity planning and monitoring. AGENDA sets up the work schedule for up to ten days of operations, automatically puts the tasks into sequences and monitors their parallel progress in real time. It works all over the network, using a "Master / Slave" protocol. Monitoring is centralized on the master computer, while the tasks can be run anywhere. It allows operator intervention at any time. In case of failure, it offers an automatic application recovery. To guarantee its perenniality and evolutivity, it relies upon de facto standards and a middleware layer.

AGENDA has been choosen because its **visualisation function** is typically concerned with **standard access**. Indeed, everybody implicated in the operational control center exploitation should be interested in the global activity visualisation around satellite passes. The problem on the existing application is that the number of visualisation monitor (X terminal) is limited by hardware architecture constraints

The selected technology : JAVA

Java has been choosen because this technology allows to reach both objectives :

- **Standard run procedure** : a Java applet can be run easily, through any web browser including the Java Virtual Machine (JVM) ; it can thus allow a user-friendly, flexible and ergonomic access point,
- **Hardware platform independance** : Java is **portable** as the pseudo-code generated is readable by any web browser including the JVM on any kind of hardware platform (PC, Workstation,...).

Moreover, it has interesting advantages :

- **object oriented** language : it offers the advantages of the object technology (i.e.supports encapsulation, inheritance and polymorphism) and that facilitates the code legibility and its maintenance.
- **distributed** : it allows to unload the server node as the applet executes on the client node. Java allows access to distributed information at application level as its API provides access to standard networking protocols (HTTP,FTP, TCP/IP) and their features.
- **simple** : there is no pointer management, and that limits confusion and bug generation.
- **robust** : Java code is checked both during the compile phase and at runtime ; its memory garbage collector avoids problems of memory allocation and memory release to the programmer.

- - **secure** : the Java interpreter verifies that the loaded pseudo-code respects the JVM security specifications in order to guarantee the system integrity.

Software Development

Development phases

The software development was composed of four principal phases:

- requirements analysis (0,5 month),
- Agenda Graphical User Interface (GUI) mock-up (1 month),
- design (1 month),
- coding and validation (2,5 months).

Design

Constraints and design choices

The main constraints we had to take into account during the design phase were :

- no sources modification of the operational software,
- no disturbance of the operational software,
- exactly the same GUI as the existing one.

The two first constraints were easy to respect as the control center Operational Activity Description (OAD) is periodically saved in an ASCII file (in order to allow the automatic application recovery), and this OAD file is available on an NFS exported partition.

Then, the main design questions we had to focus on were :

- "Which mechanisms to broadcast the information to visualize, from the WEB server to the Java clients ?"
- "Will the Java GUI development toolkit (i.e. AWT : Abstract Windowing ToolKit) allow us to get the same X11/MOTIF Agenda GUI ?"

The figure here after, shows the OAD information distribution mechanism implemented on the Java prototype.

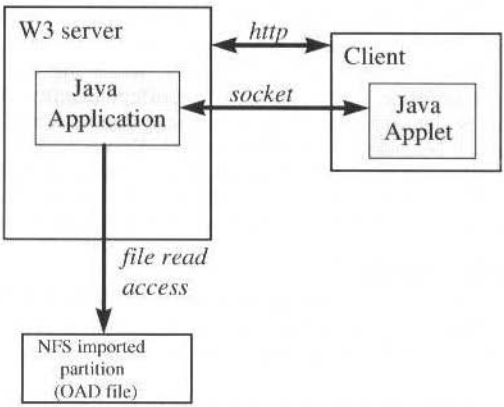


Fig. 1 OAD information distribution mechanism on the Java visualisation prototype -

Static analysis : classes overview

The method used for software design is OMT (Object Modeling Technique).

The Fig. 2 gives both an outline of the visualisation function GUI and the links between this GUI and the Java classes.

The Agenda window is divided in four main areas :

- *TextArea* : for logbook messages,
- *PGTPanel* : for operations tasks visualization (cf description in § 1.2)
- *StringPanel* : for tasks sequences visualization (cf description in § 1.2)
- *BARcanvas* : for visualization control functions (eg : zoom,...)

The *PGTPanel* and *StringPanel* objects are respectively composed of the *ProcPopupLabel* and *StringPopupLabel* objects (which are respectively the representations of a procedure, i.e. an operational task, and a procedures chain). *PGTPanel* contains, moreover, the representation of the "Universal Time" reference mark (Canvas), the mobile reference mark (RepereCanvas), the satellite passes (VisuPassage), and the delimitors of chains (Canvas).

To centralize the events management and the methods which modify the display, the *ProcessEvents* class was created. It is implemented by *FrameAgenda* and its reference is transmitted to *BARcanvas* and *PGTPanel*, as well as to the scrollable object which contains *PGTPanel*.

The main classes design is represented in the Fig. 3 :

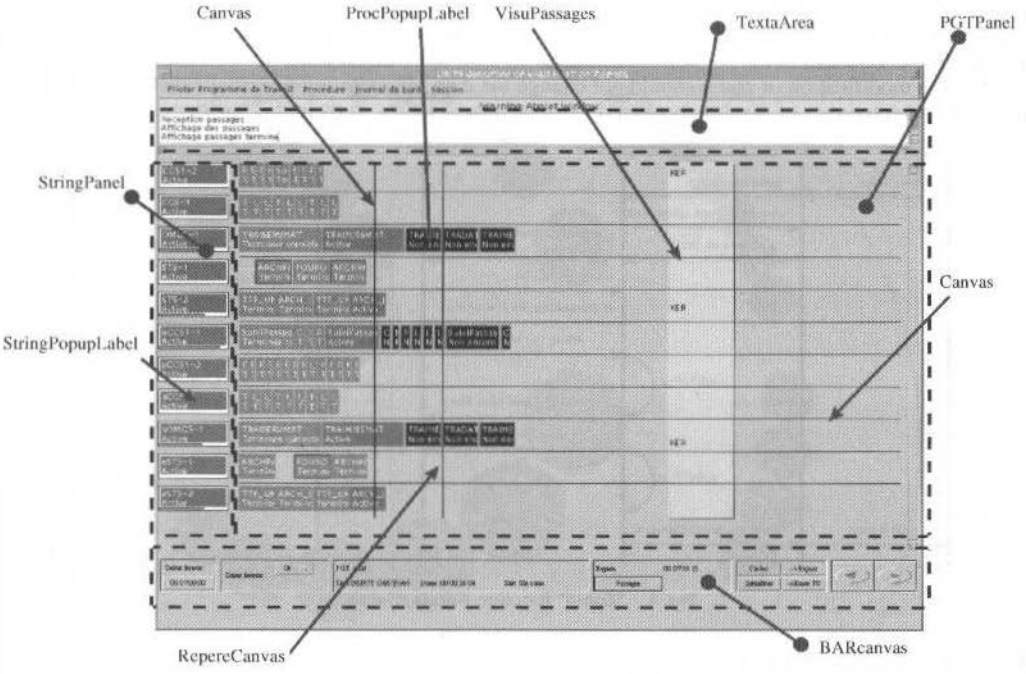


Fig. 2 Links between the prototype GUI and the related Java classes -

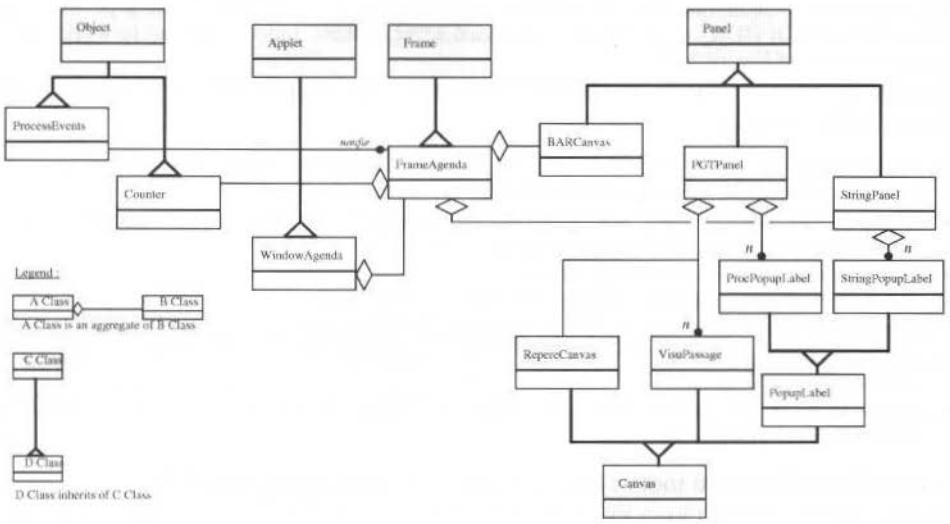


Fig. 3 General view of main classes -

Dynamics Analysis : an event diagram example

The Fig. 4 gives an outline of the diagrams created for the dynamic design of the visualization function.

It highlights the various classes implied in the the "zoom" function implementation as well as the various events exchanged between these classes.

This Fig. describes the applet reaction to a user request for visualization duration (i.e. "zoom") modification. This choice is made in a *Choice* object pertaining to *BARcanvas* class. As indicated before,the *ProcessEvents* class plays the role of a

"listener" for many objects. It is the case here and *ProcessEvents* receives the zoom modification event. It asks *BARcanvas* the corresponding duration in seconds. Then, it asks *PGTPanel* the display characteristics and activates the zoom in the window. This zoom consists, for PGTPanel, to apply a multiplying coefficient to the X-coordinates on the objects to be displayed.

Since a modification of zoom should not change the date of beginning of visualization, *ProcessEvents* uses the characteristics of display recovered to replace the window in its initial position.

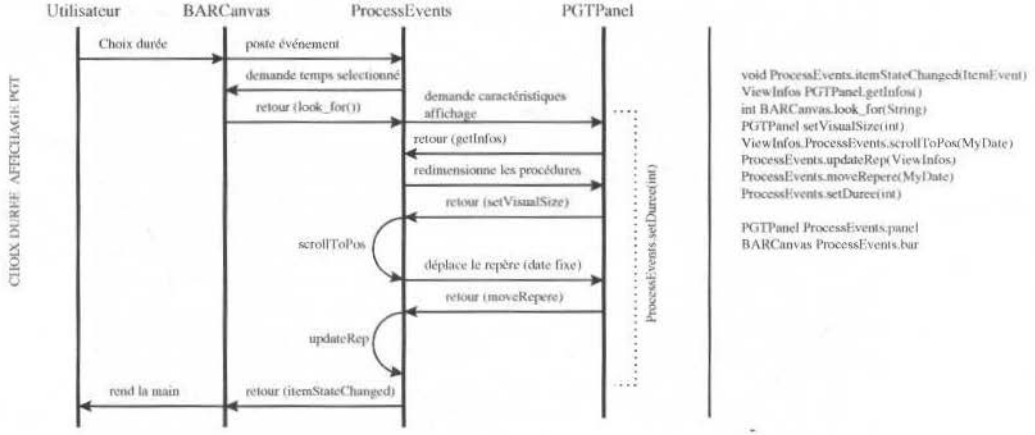


Fig. 4 "Zoom" function events diagram -

Coding and Validation

The Java Development Kit

The coding and validation phases have taken place from may to july 97, during a period when the Java Development Kit (JDK) used has been upgraded several times. Indeed, we had to work with three differents versions (cf Fig. of §4.1).

Certainly due to its youth, we encountered some problems with the JDK among which the following lacks or bugs :

- the difficulty of having up to date information (the JDK is a free product which has evolved very quickly and without support at this moment),
- *popupMenus* always have the same background color as their father one (it misses the *setBackground(color)* and *setForeground(color)* methods),
- if a ' because *Canvas* are displayed over,
- the *ScrollPane* size (highth x width) is limited to 32000 pixels.

On the other hand, we have appreciated the following positive points :

- the effectiveness of the exchanges in the Java mailing lists and Web forum of the Internet community,
- the usefulness of freeware tools like **hyperprofiler**, a reverse-designing tool (cf §2.4.1) or **mosha**, a decompiler tool.

Example of Java freeware tools : hyperprofiler, a reverse-designing tool

Among the Java freeware tools we can find, we have used "Hyperprofiler", a user-friendly reverse-designing tool.

From the Java execution logs (with -prof run option), hyperprofiler tool can build, on a Poincarré sphere projection, a tree which leaves stand for classes methods.

On the Fig. 5, we can see an example of representation obtained with Hyperprofiler.

On this graph, all the methods called are represented. HyperProfiler does not allow, in the tested version, to select the methods to be represented. However, it is completely possible to directly treat the profile file generated by the interpreter, to remove there the methods which do not interest us.

Once the graph obtained, one can change the center of projection to "zoom" parts, or click on the methods to reach their calling and called methods.

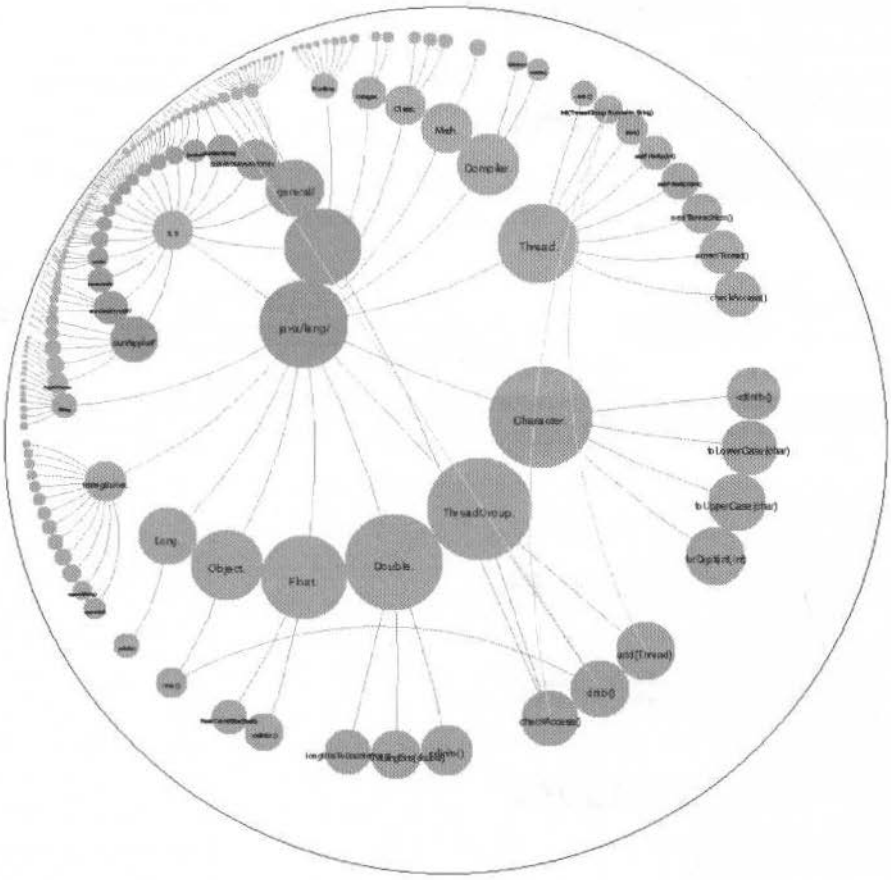


Fig. 5 An example of HyperProfiler diagram -

Results

Comparison between Agenda Motif / Agenda Java

As we can see it in the following screen hardcopies (Figs. 6 and 7), the objective which was to obtain the same GUI with Java as with Motif was achieved.

These hardcopies represent, respectively, the Motif Agenda GUI and the Java one, both while running.

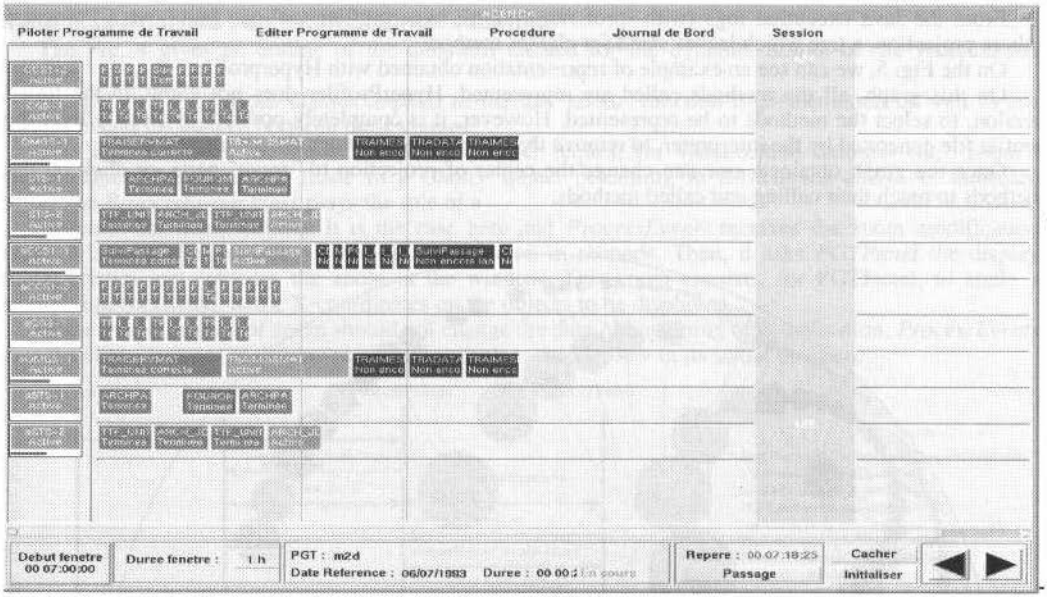


Fig. 6 The MOTIF Agenda GUI -

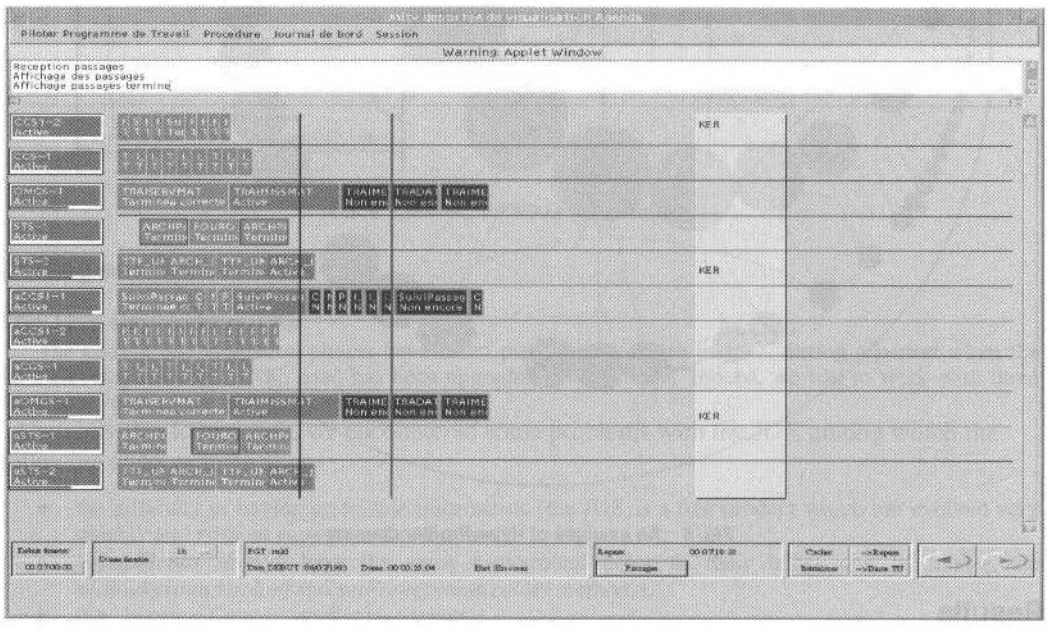


Fig. 7 The Java Agenda GUI -

Development WORKLOAD

The Java AGENDA is divided into forty classes, which represents approximately 5500 lines of code. The development workload, in which is included the training of Java language and the learning of AGENDA specifications, took 5 months (including 2,5 months for coding and tests).

Code Volume	Workload (Men/month)
5500 lines (40 classes)	4

Conclusion

Java Technology

Among Java technology benefits, some were particularly appreciable for the development phase. We can note the following ones :

- **Object** language : the code legibility is really improved by the fact Java is a "full object" language ; this is helpful for code second reading and maintenance,
- **Interpreted** language : debugging is easier because most of coding errors are quickly detected during the pseudo-compilation. If an error occurs during the program execution, the JVM gives helpful informations for the diagnostic.

We found two main problems :

- **performances** : Java is much slower than a compiled language like C but this may not be a problem if we consider that CPU powerness is nowadays not so expensive (in fact, this may be a hardware architecture choice problem),
- JDK **fast evolution** : certainly due to its youth, the Java Development Kit was relatively instable during the Java Agenda development (from march 96 to july 97). But JavaSoft, the Java software editor, has announced the JDK stabilization from the 2.0 version.

Agenda in Java

As a conclusion, we should say that the results are positive as the Agenda visualization function prototype developped in Java works correctly and the two main objectives have been reached :

- offer a **standard access** to an already existing data processing application,
- verify that the **Java** GUI development toolkit (**AWT**) is as rich as X11/Motif one.

Beyond the successful results exposed above, the **originality** of this **practical experience** consists in the combination of the concepts used : the standard access concept and the **refurbishing** one, both implemented with the Java technology. Moreover, that allows to benefit from the advantages of the object oriented technology on already existing data processing applications.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 175-182

ISSN 0100-7386
Printed in Brazil

JSWITCH/JSAT: Real-Time and Offline World Wide Web Interface

Abigail H. Maury
Computer Sciences Corporation
7700 Hubble Drive
Lanham-Seabrook, Maryland 20706 USA
e-mail: amaury@csc.com

Anna Critchfield
Computer Sciences Corporation
acritchf@csc.com

Jim Langston
Computer Sciences Corporation
jlangsto@csc.com

Abstract

Jswitch, a Java-based spacecraft Web interface to telemetry and command handling, is a prototype, platform-independent user interface to a spacecraft command and control system that uses Java technology, readily available security software, standard World Wide Web (WWW) protocols, and commercial off-the-shelf (COTS) products. Jsat, a Java-based science analysis and trending tool, is a major element in Jswitch. Both Jswitch and Jsat were developed for NASA's Goddard Space Flight Center (GSFC). Jsat provides Web interface, user access to science instrument data by sending the processed science satellite trend data (user-specified graphics, tables, and reports) as Java applets via the Web to the user upon request. Jsat reduces ground science instrument data processing costs and decreases processing time by eliminating the need for special operations personnel assigned to process science instrument data requests, run trend analysis software, and collect and distribute results. Jswitch reduces operational costs by providing access to spacecraft telemetry and command from any standard Web browser.

Keywords: *Spacecraft Web Interface, Telemetry and Command Handling Web Interface, Science Data Trend Analysis Web Interface.*

Introduction

NASA's Goddard Space Flight Center (GSFC) is funding a research effort in the use of the latest technology to reduce satellite ground system operations costs. This effort has produced the Jswitch/Jsat system, which enables a spacecraft/flight operations engineer to monitor and control a spacecraft, or enables a scientist to select and analyze the spacecraft science instrument data, over the open Internet using the standard World Wide Web (WWW) tools, browsers, and commercial-off-the-shelf (COTS) products, in conjunction with the Java programming technology. Encryption, certification, firewall, and intrusion detection technology provide security.

Jswitch implements a generic mechanism by downloading applets to access mission control center systems over the Internet. The Jswitch system interfaces through a generic bridge to COTS control center systems. For example, we process telemetry pages through an interface to the following systems: EPOCH 2000, Transportable Payload Operations Control Center (TPOCC), Generic Spacecraft Analyst Assistant (GenSAA), and the Advanced System for Integration and Spacecraft Test (ASIST). Jsat, a major element in Jswitch, interfaces with the Archive Browser Extractor (ABE), a COTS non-real-time spacecraft telemetry data analysis package that provides the statistical subset data. Integral Systems, Inc. (ISI) developed EPOCH 2000 and ABE and provided both products to NASA/GSFC for prototyping the Jswitch/Jsat system. Currently, GSFC is integrating components of the Jswitch system into the OMNI (Operating Mission as Nodes on the Internet) system for the Solar Eclipse Mission to the Black Sea in August 1999. OMNI will transmit weather, Global Positioning System (GPS) data, and eclipse images to museums, schools, and the public over the Internet.

The Jswitch/Jsat system demonstrates how a combination of current technologies can be used in mission support systems today. The Jswitch system reduces operations costs through operational scenarios that enable remote diagnosis of problems detected during lights-out operations periods. The Jsat system removes the necessity for operators to provide data to scientists by giving scientists direct access through the Web. The system allows distributed operations teams to use the inexpensive open Internet for wide area access.

The Jswitch/Jsat system provides flexibility for use across many mission profiles by using COTS products to reduce development costs. The net result is the first opportunity to standardize user interfaces to missions, provide flexible networking, reduce operations, and significantly decrease life-cycle costs.

Jswitch/Jsat Advantages

The Java-based Jswitch software interfaces with EPOCH 2000, TPOCC, GenSAA, ASIST, or other ground systems through an application programming interface (API) using C code or Java. Using a distributed architecture, the control center software can be replaced relatively easily while the Jswitch user interface stays the same. The Jswitch/Jsat prototype gives the user the ability to view events and telemetry data, send commands, and analyze selected statistical science data—all through the use of a Web browser as the user interface. From our experience, this approach offers a number of advantages:

1. Distributed workload. Once the Java applets are downloaded to the client machine from the connected host, the applets run locally on the client's machine.
2. Portability. The Java applets will run on any machine that has a Web browser that supports Java—without any required source code changes for a specific platform (one version for all).
3. Easy user access. There is no complicated setup involved or additional software to buy for clients—the only requirement is a Web browser that supports Java and an Internet connection.
4. Easier software maintenance. Clients always connect through the central Web server, which automatically downloads the applets, making it easier to install updates to the software.
5. Direct access to the data by the scientific community. Using Jswitch/Jsat security mechanisms, user privileges can be set up to allow direct access to data for analysis without jeopardizing command and control security.
6. Availability of legacy telemetry displays via x-server Internet plug-in. The original display definitions are unchanged.
7. Generic control center interface. All mission interfaces have the same look and feel from a single Web page.

Jswitch/Jsat Web Security

Security over the open Internet is recognized as the major risk in satellite operations, particularly in commanding the satellite. Jswitch employs a number of readily available security tools and procedures to provide a secure interface. The heart of the secure interface consists of the Stronghold Web server, which is a Secure Socket Layer (SSL) implementation of the Apache Web server. Netscape's SSL, a de facto standard now, encrypts all transmissions, making it difficult for the data sent between the user's Web browser and the Web site to be monitored in transit. The encryption strength used depends on the Web browser version installed by the user. The Web server's function is to serve Java applets to the user. These Java applets provide for a graphical dynamic interface to the ground system. The applets connect to data servers to send/receive requests and data to interface programs that talk to the actual ground system. The Java applets, once downloaded on the client's machine, will run locally—only connecting back to the host for data and/or command information. The Stronghold product is easy to use, which is important because errors in complicated setups could compromise the use of the product.

This approach, as installed in the Jswitch EPOCH 2000 prototype, is to pass the data packets through the Web server and use the Web server's encryption capability (see Figure 1). The key to using server/browser encryption is to provide the data to the applet in a standard output (for telemetry) or standard input (for commands) stream via a cgi script. For data moving from the server to the browser, the applet asks to get a stream object (i.e., creates a connection to the standard output of a cgi script) using a statement such as:

stream = URL "https://
jswitch.nascom.nasa.gov/NPH_event.cgi ? sim1"

The standard output stream is established and comes from the NPH_event.cgi script. The Web server encrypts because the https:// directive is used. The parameter sim1 is used in our configuration to pass to EPOCH 2000 to ask for the data source sim1. The NPH_event.cgi script contains a call to event.c that in turn connects to EPOCH 2000. After the applet establishes the stream, the applet iterates references to stream.getline() to retrieve each packet of data that is encrypted by the server and decrypted by the browser.

The SSL Web server also provides support for digital identification, used to help assure that clients are connecting to the actual Web site and not a "bogus" site. The Jswitch Web site uses a digital certificate from Verisign. We employ a form of "firewall" software on our Jswitch machine called TCP Wrapper. This freely
available tool acts as a "wrapper" for common Transmission Control Protocol (TCP) services. It provides for more detailed logging and allows for a host access and deny list for TCP services (ftp, telnet, etc.), all of which are traditionally used for remote access of a machine.

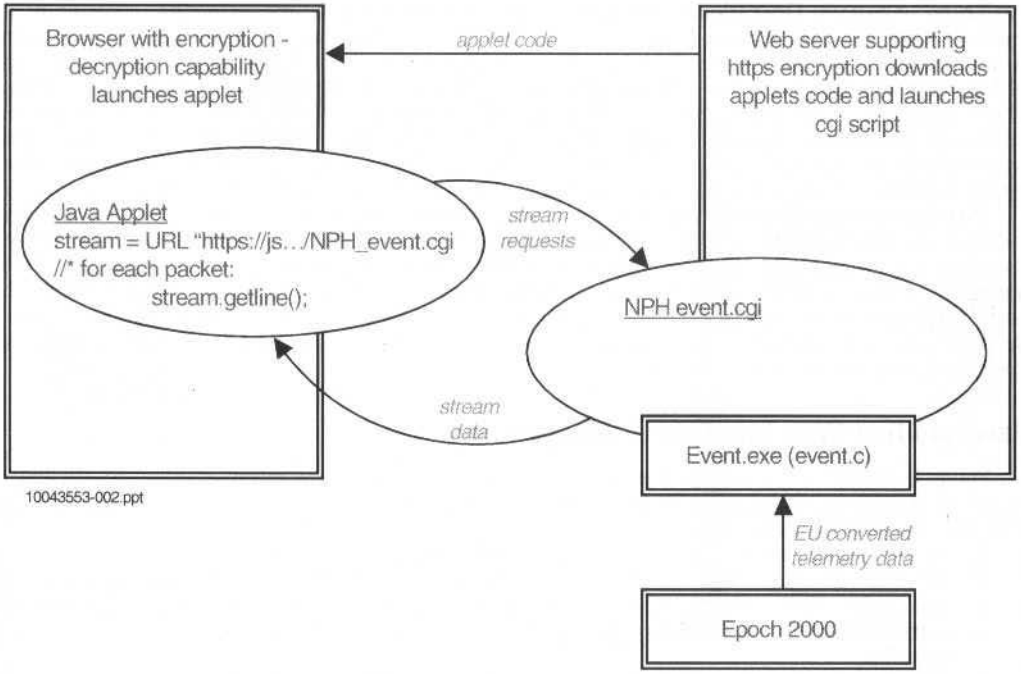


Fig. 1 Encrypted_Streaming_Objects Architecture

The Jswitch machine employs a form of intrusion detection software called SWATCH, freely available on the Internet. The SWATCH tool monitors the system logs and watches for specified patterns that indicate some potential foul play (such as login attempts from unauthorized users). Free probing software, called SATAN, used to check the host machine for weaknesses, was installed on another host to scan the targeted Jswitch machine for security weaknesses. Additionally, alteration of some system configuration files and services on our UNIX machine has led to a considerable improvement in system security. The primary modifications involved disabling services that are enabled in the Internet configuration file (/etc/inetd.conf on most systems).

Finally, the Jswitch machine configuration itself offers a simple, easily maintainable security solution. All the software and hardware reside on one standard UNIX machine, connected to the open Internet, but the host machine could easily be set up to have no connection to any other internal network. The only needed connection is for the ground system software to access a socket, which has the connection to data or the actual front end of a ground system. In this way, the data or front-end machines can still be kept behind whatever firewall in which they currently reside.

Operational Scenarios

By providing a uniform user interface and security mechanism, the Jswitch/Jsat architecture supports all typical operational scenarios scripted for users with different levels of operational privileges. Jswitch/Jsat naturally integrates into GSFC's use of multiple redundant strings to support satellite contacts. The Jswitch concept supports important operational scenarios that are needed to move closer to reduced operations staffs and lights-out operations during some shifts and to eliminate dedicated operations areas for small missions, enabling operators to work from their desktop computers. The following paragraphs describe some typical operational scenarios.

Anomaly Resolution Scenario: The ground system, during non-staffed off-shift hours, receives data and performs data analysis. If the ground system detects an anomaly, the system phones or pages operations personnel and/or engineers. The operations personnel from their home or office connect to the ground system over the Internet to diagnose problems by examining telemetry from recent or current passes. The operations personnel prepare and issue a real-time spacecraft command or directive that will be sent to the spacecraft during the next contact to resolve the anomaly, maintaining spacecraft health and safety.

Trend Analysis Operational Scenario: The intended user is the scientific community or engineers/analysts. User privileges are limited to playback and analysis of history data on the offline string. The user logs into the Jswitch system; the system recognizes limited privileges and connects to the offline EPOCH/ABE string. The user requests generation of statistical analysis of specified mnemonics. The Jsat component passes the request to ABE, and ABE returns the results to the user over the Internet in textual or graphical dynamic displays or formatted reports. This example shows how easy trend analysis becomes when using Jsat, compared to a legacy control center that involves time-consuming subset generation by the operations team and then distribution of correct subsets to multiple users.

Distributed Operations Scenario: Ground system servers are located in a closet at a convenient location for configuration control and hardware maintenance. Backup system(s) could be located elsewhere for geographical security. Operations personnel and spacecraft engineers perform operations duties from their desktop PCs in their offices during normal work situations and hours. Operations can be performed from remote sites during non-typical situations (e.g., conferences or meetings at other centers). Scientists and other researchers can perform science planning, receive science observations, and analyze this information from their own offices.

Jswitch/Jsat Web Interface Architecture

Figure 2 illustrates the Jswitch/Jsat architecture. ABE was the most recent addition to Jswitch to create a Jsat component that conceptualizes direct user science instrument data processing and analysis. This Web interface works as follows:

1. A Web server runs on the application host machine for remote clients to access.
2. Data server applications running on the application host machine receive socket requests from the applets and send the response back, via a socket, to the connecting Java applet. Another version uses the server encryption to return data packets via a cgi script/standard output interface.
3. The user interface is a Web browser (like Netscape Navigator) in which the user enters the Uniform Resource Locator (URL) for the Web site. This will produce the Web page and download the Java applets from the server to the Web browser on any platform. (Note: Applets are not reloaded when they are already in the browser cache.)
4. The Java applets pop up in their own frames, allowing users to enter commands, view real-time event data, monitor telemetry data, and talk to each other.

The EPOCH 2000 system and the ABE data extractor are integrated with other COTS products as a mission operations control server running on the UNIX workstation. COTS command and control systems, for example, EPOCH 2000, provide real-time playback or simulated telemetry and events in the spacecraft downlink to the Jswitch applets. The commands for uplink are received by EPOCH 2000, which performs the criticality and constraints checking and subsequently uplinks the commands to the spacecraft. The Jswitch Java code interfaces with the EPOCH 2000 system API (C interface routines) for telemetry, command, and events. The Jsat system will use a standard text editor to generate a profile file that contains all of the information that the ABE needs to generate an ABE trend table and store the data in a file for retrieval by the Jsat system. A Jsat applet retrieves the data for display on any platform with a Java-enabled browser. The Jswitch/Jsat system has a current set of tools for encryption, certification, and authentication to protect client-side security, server-side security, and document confidentiality.

COTS Products

Jswitch Software: The current COTS products in the Jswitch prototype suite are as follows: Stronghold Web server by C2net (http://www.c2.net), Digital id by Verisign (http://www.verisign.com), Java Development Kit (free, http://www.javasoft.com/products), Jswitch Java source code (free, http://moca.nascom.nasa.gov), SSLava Toolkit by Phaos (http://www.phaos.com), ground system interface software (free), and Jchart library by KLG group (http://www.klg.com.jclass/overview.html).

Software and configurations to make the client machine more secure include the following: TCP Wrapper, SWATCH, SATAN, SSH, modifications to /etc/inetd.conf services, and Secure ID by Security Dynamics (http://securitydynamics.com). These are downloadable from various sources on the Internet; one central source is ftp://coast.cs.purdue.edu/pub/tools/unix/.

The COTS control system is EPOCH 2000 by ISI. Legacy, custom control systems are TPOCC, GenSAA, and ASIST.

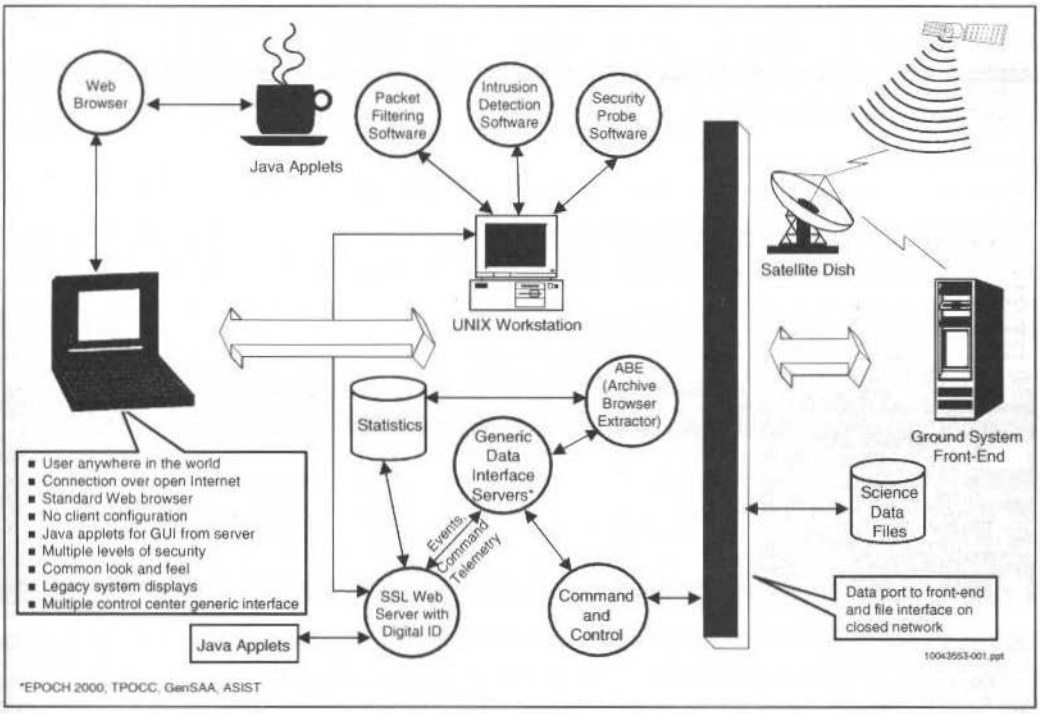


Fig. 2 Java-Based Spacecraft Web Interface to Telemetry & Command Handling

Jswitch Hardware: Clients can theoretically use any Java-aware Web browser. Due to differences in some vendors' Web browsers, we find that Netscape Navigator 4.6 works the best. This is free and can be downloaded from http://www.netscape.com. The host machine, needed to serve the Java applets, run the Web server, interface to the ground system, and run the interface servers, has been Sun's UltraSparc 2, running the Solaris 2.6 operating system. The host machine does not need a Sun box as long as it is running some form of UNIX. The only caveat is that the machine should have an adequate amount of memory and processing speed to handle all the connecting clients.

Jswitch/Jsat Implementation

A Web server runs on a host machine to provide access to clients. Users, from the server's Web page, opening their Web browsers and connecting to the host site, can select which Jswitch/Jsat functions to invoke (command, events, telemetry, or trend analysis). Selecting a function will invoke a Java applet, which will be downloaded to the client machine and then run locally. The Java applet will pop up a window, providing a user interface for the specified function. The applet will make a socket connection back to the host machine to send and receive data from a data server that is running on the host machine.

Jswitch

Figure 3, a screen snapshot, shows the Jswitch command, telemetry, and event applet running on a PC with the selected telemetry plot applets displayed. The command applet provides the user with a text entry field for entering commands to the ground system, as well as a menu selection of common commands that can be inserted by clicking. Once a command has been entered, clicking on the send button will send the command to the host machine, where the data server will send the command to the ground system, get the result back, and then return it to the command applet. The command applet will display the original command, along with the result, in a scrolling text area. The command action is entered into the scrolling event display.

The event applet provides the user with an event list from the ground system. Once activated, it will connect to a data server, which fetches events from the ground system. These events are displayed in the

event applet in a scrolling text area. Event messages are color-coded (red, out of limits; yellow, starting to exceed limits;green, normal limits; and white, spacecraft events). The user can filter events.

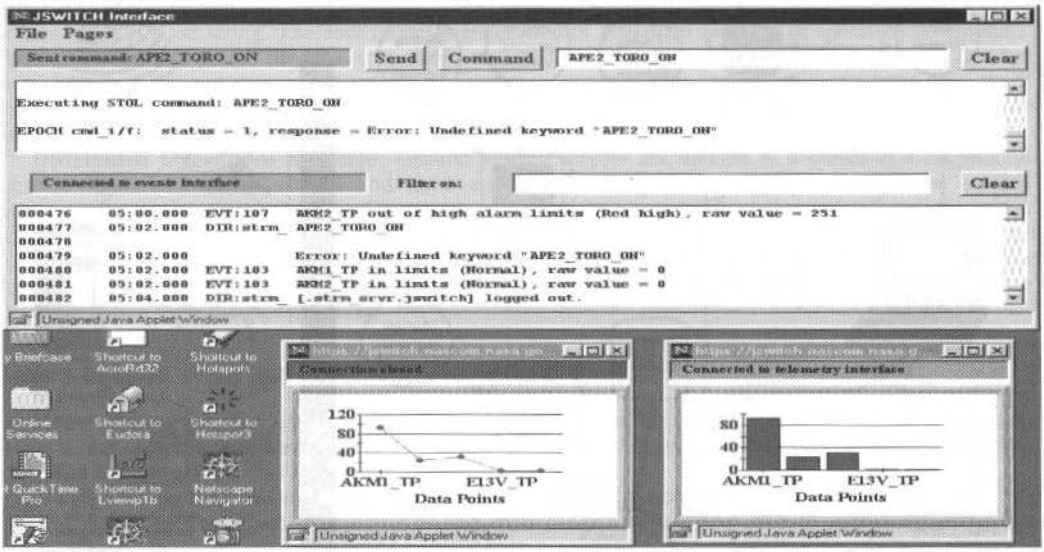


Fig. 3 Jswitch: Telemetry and Command Applet

The telemetry function provides the user with a list of directories and pages to select. These pages are American Standard Code for Information Interchange (ASCII) text files residing on the host machine that describe the appearance of a telemetry page and a list of data points. Currently, the displays consist of basic text displays and graphical displays. Additional graphical displays are available by using COTS products, which provide Java graphical libraries. Once the user selects a desired page, this information is sent to a data server, where data updates are sent to the telemetry applet, which displays the telemetry information in a window. The users may define their mnemonics for graphical display.

Jswtich/X-Protocol Display Enhancement: Jswitch has added an option to make viewing of the Gamma Ray Observatory (GRO) legacy telemetry pages via an x-server Internet plug-in running on a desktop PC. Historically, spacecraft mission telemetry pages are customized for each spacecraft, and conversion to a different format is costly, involving retraining operations personnel. By providing legacy mission displays with no changes, operations can continue with no impact and without extra costs, and the chances of operational errors associated with changing existing displays are reduced.

Jswitch/Generic Bridge Enhancement: Jswitch is developing a common interface to connect to multiple control centers. This approach has been demonstrated to work in the connection to EPOCH 2000, TPOCC, and the GenSAA toolkit. We are currently prototyping connections to other mission systems such as GSFC's ASIST. The common Web interface will present multiple control center interfaces associated with privileges to access the data for operations personnel and schools.

Jsat

Jsat interfaces with ABE, the ISI COTS spacecraft telemetry data analysis package that provides the user with a set of browsing and data analysis functions for retrieval and decoding of actual telemetry data into statistical subsets of user-defined telemetry data. The ABE package is hosted on the server and is usually associated with EPOCH 2000, allowing access to spacecraft databases for the retrieval of telemetry data, event, and trends data files. The ABE package uses PV-WAVE; a commercial, general-purpose, visual analysis package from Visual Numeric, Inc., PV-WAVE is widely used in the technical community as a foundation for data analysis.

The Jsat trend display is shown in Figure 4. Clicking the Trend menu will launch a graphical user interface (GUI) applet with selectable directories and data subset files. The selected data statistical subset may be viewed with a "View Source" button prior to downloading. The user may select from the following data subset display options: text, plot, scatter plot, bar, area, etc. The statistical data is sent to the text display functions or to a chart in a chart library commercially supplied by the KLG Group for continuous update until the end of the data file.

The ABE remote interface consists of a simple ABE command interface routine written in C. This remote interface processes a profile file that can be generated with any text editor. The file will tell ABE to write the data to a text file to be processed. If the user has limited access only to an offline string, the applets later can be replaced by applications to enable the user not only to view but also to store and manipulate data locally on a user computer.

Platform Independence

The Jswitch/Jsat client has been tested on PC, Macintosh, HP, and Sun platforms. Netscape Navigator Gold 3.0, Netscape Navigator Professional Edition 4.04 and 4.6, and Internet Explorer 3.0 browsers have been used. Browser differences were noted in each instance with the rendering of the applets. Differences consisted mainly of font size, widget sizes, and colors. We have standardized by using Netscape Navigator 4.6 for development purposes. On the PC, the primary development platform, these differences were traced to individual preferences in control panel settings.

The colored event applet rendered true red, yellow, or green event colors in Netscape Navigator Gold 3.0 with the Color Palette setting at "256 colors". However, Netscape Navigator Professional Edition 4.04 rendered the colors incorrectly with the "256 color" setting and correctly with the Color Palette set to "high color 16 bit".

The trend analysis applet was rendered with the last column in a second row underneath the label column on the developer's PC and correctly on a second PC with all columns in one row. Turning the screen size on and off corrected this rendering on the developer's PC.

One exception was noted in the execution of code between the Java Virtual Machine on Internet Explorer 3.0 and Netscape Navigator Professional Edition 4.6. On the Internet Explorer 3.0 Java Virtual Machine, the data values were updated correctly in the labeled text fields; however, the time was

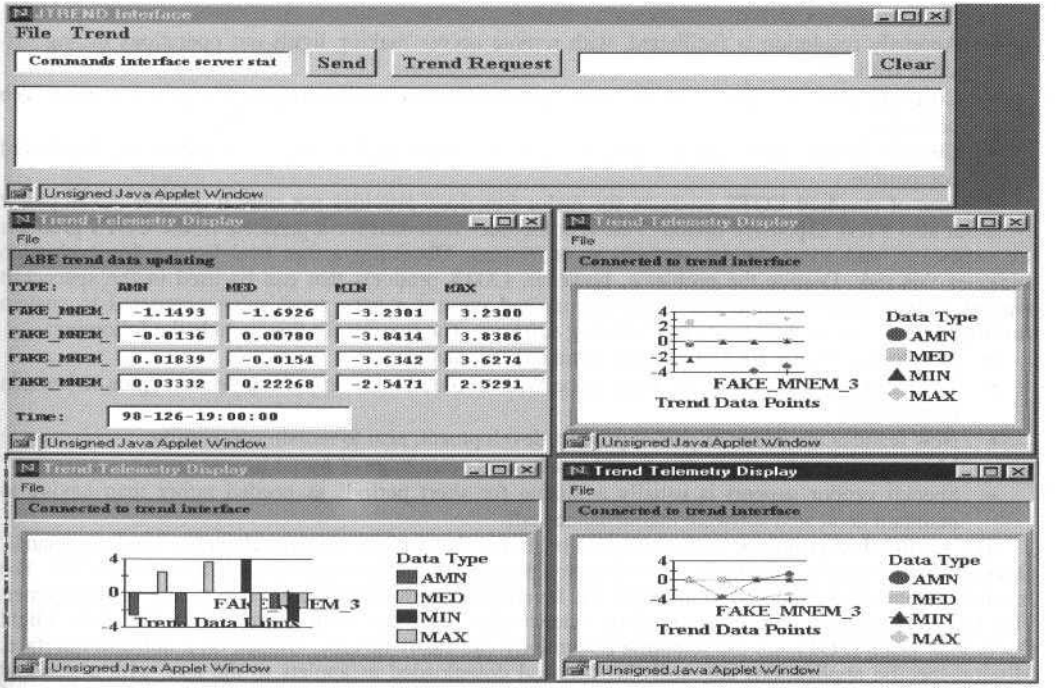


Fig. 4 Jsat: Trend Applet

incorrectly updated in the Time text field. Netscape 4.6 always displayed the time correctly. The Internet Explorer 3.0 Java virtual machine has incorrectly interpreted the following statement: if (datapoint.time != ("XTIME"). The final solution to this problem probably lies with the Java virtual machine as an add-in to the browser. This product is under development by Sun.

Summary: Jswitch/Jsat fits very well with inexpensive small satellite rapid development

Jswitch/Jsat can quickly provide reusable ground software, with the user evaluating the software from his/her location and providing immediate feedback to the developers. Scientists have a quick, simple access to process their science data. Building ground systems faster is achieved first by using existing software and second by integrating the system using an iterative development process. In the Jswitch prototype systems, several existing COTS products were used. Core command and control functions are readily available in several COTS products (e.g., ISI EPOCH 2000, STI OS/Comet, Altair MCS, LM SCS 21) and Government off-the-shelf (GOTS) products (TPOCC, ASIST). The GenSAA toolkit provides highly graphical expert systems that perform real-time spacecraft monitoring and display. To make use of the WWW, COTS Web servers and browsers were examined that include security features and support the Java virtual machine. The C2net (Apache) Stronghold COTS Web server provides security through encryption and authentication through passwords.

The Jswitch concept enables a better ground system by providing the definition of standard GUI elements that are based on the Java language and applets served up through the Java virtual machine under standard Web browsers. This is a powerful combination because virtually every work location has recent, free Web browsers from Microsoft or Netscape already installed. These products are available for Windows 95, Macintosh, and most versions of UNIX operating systems. In summary, the user benefits from the following capabilities:

- Scientists can communicate with an instrument using a Web browser.
- Engineers and operators can work from office, from home, or on the road using only a Web browser.
- Client computers are platform independent (Web browsers run on PC, Macintosh, and UNIX platforms).
- Anomaly resolution is facilitated, with remote access making lights-out operations during off-shifts realistic.
- Costs are lower; no special hardware or special client software is required beyond a current Web browser.
- Generic bridge interfaces to legacy control centers and telemetry displays are backward compatible.

The use of standard COTS products for major functions provides not only a quicker development cycle but also a less expensive option. This is sometimes mistaken for a panacea. Some COTS products are expensive, and some do not perform as advertised. Obtaining a free trial license for evaluation mitigates this risk. Despite the problems, however, COTS products that can be used in the spacecraft support domain have matured over recent years and present a very good value compared to custom development or even to modification of existing software. The reasons for this include the following:

- Development costs are amortized over many customers.
- The product is enhanced to keep pace with evolution in supporting technologies to remain competitive.
- New features are added as part of product development, also to remain competitive.
- Maintenance costs are usually quoted as a known percentage of initial license costs.
- Special vendor support is usually available for short periods as needed rather than having to maintain dedicated software maintenance staff.

The Jswitch/Jsat prototype system has demonstrated that we can receive telemetry, send commands, process events, display existing legacy telemetry pages, and analyze statistical data over the Web. The Jswitch system can be tailored to interface the Java language through native code to any control center. The Jsat component can easily download science data that has been written to a file. The Flight Operations Team (FOT) and the scientific community can easily be cross-trained on a uniform interface. Mission-specific applications are easily written and downloaded as applets to the user with a standard Java-enabled Web browser. The user always has the latest software updates, and there is no special hardware to buy.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 183

ISSN 0100-7386
Printed in Brazil

Ground Systems II

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 184-190

ISSN 0100-7386
Printed in Brazil

A Compact and Autonomous Ground System for Saci-1 Mission Controlling

Maria de Fátima Mattiello Francisco
José Angelo da Costa Ferreira Neri
National Institute for Space Research -INPE/SJC
fatima@dss.inpe.br
neri@dea.inpe.br

Abstract

This paper address the Ground solutions adopted to control the scientific microsatellite SACI-1 in order to meet the operation and functional requirements. Those requirements are unattended control of a satellite passing, quick-look of payload data in real time, playback of all telemetry data, automatic orbit determination and remote mission planning by the investigators through the Internet. Since cost-effectiveness has oriented all design solutions in SACI-1 Ground Segment, a compact satellite control system was designed to aggregate in a same environment the functionality of Ground Station, Control Center and Mission Center. That compact SACI-1 Ground System is a platform based on five Personal Computers interconnected through Ethernet local network. Two of them are dedicated to perform TM and TC functions including the Ground-satellite protocol. Another is dedicated to the antenna control for satellite tracking. The last two deals with data dissemination activities such as quick look of payload data and data base server for remote controlling of both current and historical data of the mission.

SACI-1 Mission

Focusing on SACI-1 purpose, it is a low earth polar orbit microsatellite intended for scientific application, (Neri, 1996 et al.:). The SACI-1 payload concerns four scientific experiments for systematic observation of the ionosphere. SACI-1 shall be launched as CBERS (China-Brazil Earth Resource Satellite) piggyback, by Chinese launcher Long March IV, on July 1999.

Briefly, the four scientific experiments on-board of SACI-1 are:

- PLASMEX - intends to investigate the plasma bubbles generation phenomena, its development and decay over the Brazilian area.
- ORCAS - intends to monitor, in the inner magnetosphere, the fluxes and spectra of electrons, protons and He to Ne ion populations of energies bellow 100 MeV/amu.
- PHOTEX - intends to investigate the equatorial ionosphere anomaly, South Atlantic Magnetic Anomaly (SAMA) and dynamic process in the mesosphere in global scale.
- MAGNEX - plans to conduct vector geomagnetic field measurements continuously in three-axis orthogonal components on the microsatellite.

Those experiment will acquire data from the space plasma environment simultaneously during the entire mission life. The acquired payload data are compressed whenever necessary, and stored in different Payload Telemetry Application Packets (1K bytes each) for further transmission to Ground. An average of 8K packets per day from ORCAS will be received in Ground; 12K packets per day from PLASMEX; 6K packets per day from MAGNEX and 6K packets per day from PHOTEX. The payload telemetry plus the satellite telemetry will be transmitted to Ground Station in 500 Kbps rate (an average of 60.4 packets per second).

Matching the satellite orbit with the SACI-1 Ground Station position (at northeast of Brazil), the satellite visibility shall occur twice a day, in 5-8 minutes passes, with an average of ~18 Mbytes per pass. As a consequence of both the SACI-1 short contact and the high data payload acquired by the on-board scientific experiments, the Ground-Satellite protocol communication has been robustly implemented and, essential autonomous features have been incorporated to the TM/TC application software.

Ground System

A dedicated and compact Ground System (Fig. 1) was designed to control and receive data from the SACI-1 mission. It is a platform based on five personal computers, under Windows NT, Windows 95 and Linux operating systems, interconnected through Ethernet local area network.

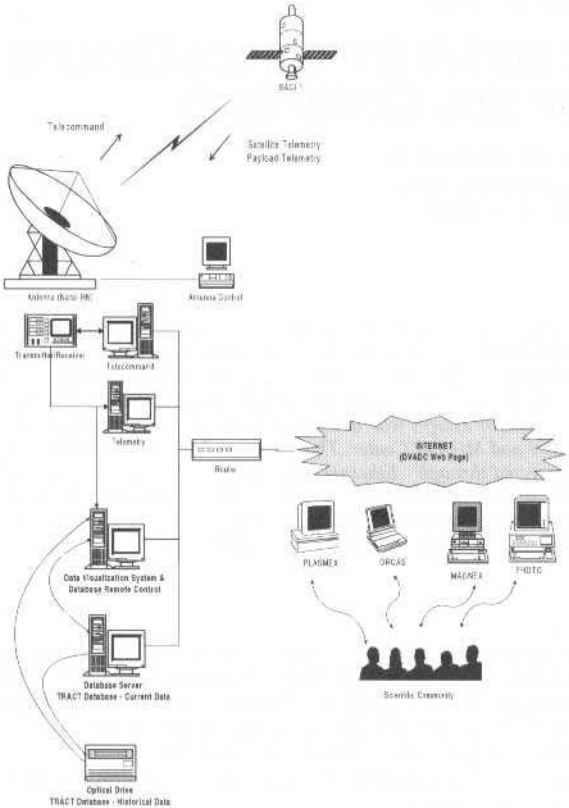


Fig. 1 SACI-1 GROUND SYSTEM

Two computers are dedicated to perform respectively Telemetry (TM) and Telecommand (TC) functions, interconnected by a serial link RS-232 which supports the implementation of the ground-satellite protocol communication. Another personal computer is dedicated to the antenna control based on the satellite orbit preview calculation. The last two deal with data dissemination activities such as quick-look of payload data and database server for remote controlling of both current and historical data of the mission.

The personal computers dedicated to Telemetry & Data Visualization System are provided with developed-in-house micro-controlled cards, allowing both systems to receive simultaneously the telemetry application packets transmitted from SACI-1 On-Board Computer (OBC) to Ground. Also, in the telecommand personal computer, there is a developed-in-house micro-controlled card for sending the telecommand packets.

Software

The Ground Segment is compound of three systems:

- Telemetry and Telecommand System;
- Data Dissemination System – DV&DC system
- Antenna Control and Positioning System

TMTC

It is compound of two subsystems which beside the application functionality implement the Ground-Satellite Communication Protocol. Concerning the Telecommand Subsystem, the implementation, (Carvalho, 1996 and Carvalho, 1998), of Ground-Satellite protocol followed CCSDS recommendation.

- The Telemetry subsystem is dedicated to receive and store all telemetry data transmitted from SACI-1 to the S-band TT&C ground station during the passing period. According to the

application specification, SACI-1 satellite generates two categories of telemetry messages: real time telemetry and stored telemetry (Non real time telemetry). This last one consists of both payload and housekeeping data.

- The Telecommand subsystem provides facilities for autonomous management, logging and transmission of Telecommand to the satellite, throuh the up-link equipment chain installed at the TT&C station.

The development of TMTC software system, named SACI-SOL, followed the incremental approach. The software modules were structured in layers as presented in Fig. 2. SACI-SOL was designed to operate in two Personal Computers under Windows NT operation system. One of them is dedicated to perform TC control while another is doing TM functions. A graphical programming environment was used to develop SACI-SOL software, (Mattiello et al.: 1998 and Neri, et al.: 1997). The segment of SACI-SOL resident in the TM computer was implemented in LabWindows / CVI in order to reach the following high performance requirements:

- Reception and recording of 60.4 packets per second during ~8 minutes of passing;
- CRC and sequence number checking of each received frame. In case of wrong reception, a message is transmitted to the TC computer through serial link informing it about the right sequence number waiting for.
- Identification of the CLCW ground-satellite protocol word received in telemetry frame and its transmission to the TC computer through serial link.

Since there was no critical requirement for TC transmission, the SACI-SOL segment resident in TC computer was successfully implemented in LabVIEW graphical environment. Aggregating that solution with PHP 3.0 (a server-side HTML-embedded scripting language) facilities to write web pages quickly, the following operation needs were reached:

- Unattended control of the passing in order to reduce the number of staffs in the Ground Station;
- Telecommand messages scheduled by users through Internet under control of SACI-1 Flight Plan software.

The use of those development environment was one of the main project solution adopted by the developers of SACI-SOL software due to their facilities in developing user interface.

The Figure 2 shows the SACI-SOL architecture. It consists of the following software modules :

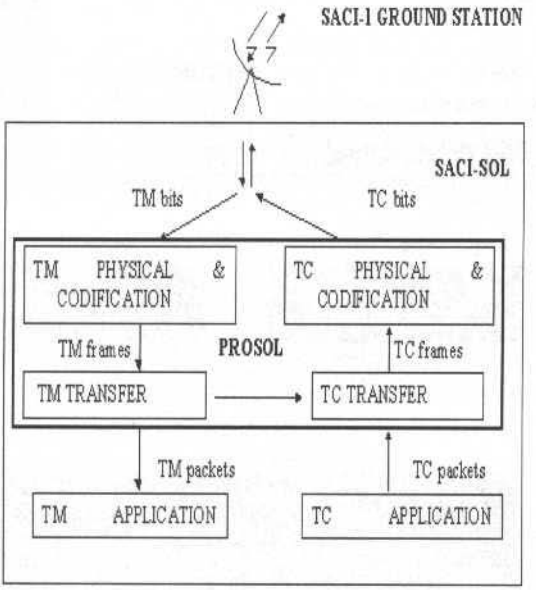


Fig. 2 SACI-SOL Module Architecture

1. *TM Physical & Codification Module* – deals with the acquisition, decodification and separation of the telemetry bits in frames.
2. *TM Transfer Module* – deals with the TM frames validation following the Consultative Committee for Space Data System (CCSDS) recommendation.

3. *TM Application Module* – deals with the high level functionality of storing and treating each TM packet received from satellite. That includes the identification of the TM packet type, the processing and the visualization of the relevant one.
4. *TC Physical & Codification Module* – deals with the codification and transmission of the TC bits to the satellite.
5. *TC Transfer Module* – deals with the treatment of the TC frames following the Consultative Committee for Space Data System (CCSDS) recommendation.
6. *TC Application Module* – deals with the high level functionality of the flight plan automatic control in order to transmit all the telecommands selected for each passage.

The design solution adopted in TC Application Module was the finite state automata[4]. In the implementation of the finite state automata, the states represent the situation in which the system stays after treating a specific event and waiting for a next one. The events are the occurrences which make the state machine change from on state to another one. And the actions are the procedures which must be executed to take the state machine to the next state.

The occurrences considered relevant for the TC automatic control were:

1. Beginning of the passage;
2. Ending of the passage;
3. Time to transmit the next TC to the microsatellite;
4. Information related to the TC local protocol services;
5. Out of sequence TM frame;
6. Information related to the TC protocol received from TM frames – CLCW word.

The Automata program keeps the state table data with states, events and actions. When the Automata starts to run, it is set to the initial state, initialize the monitors, queues and calls the Read Event process to get event. The Read Event stays running until getting an event. The Read Event returns the event to the Automata, it consults the table in order to determine what action must be called and what is the next state. The right action is executed and, after that, both the current state is changed and the Read Event is called again by the Automata.

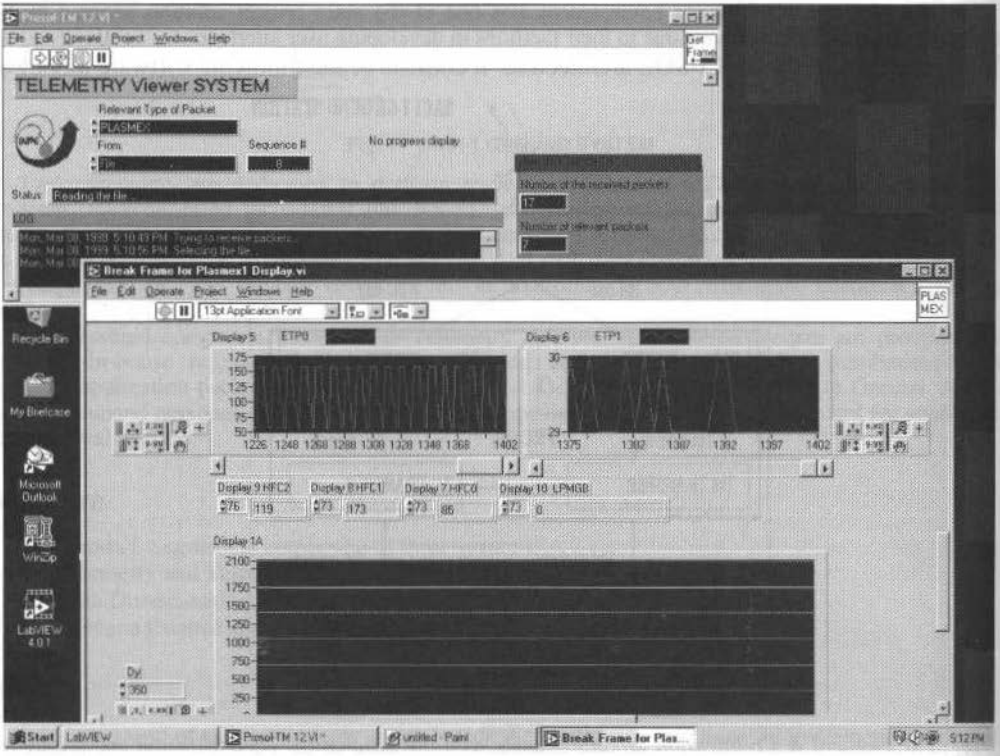


Fig. 3 Telemetry Viewer System

One of the implemented actions is responsible to transmit the next TC listed in Flight Plan to the microsatellite sending it to the TC Transfer Layer queue.

Data Dissemination System - DV&DC system

The Data Dissemination software system, Telemetry Data Visualization System & Data Base Remote Control (DV&DC system), concerns the ground subsystem responsible for SACI-1 telemetry data reception, pre-processing and treatment.

DV&DC system is a multi-user software system compounded of the following software subsystems:

- Real Time telemetry software – **RTTM** which deals with reception and visualization of the real time telemetry in real time and playback. This software was implemented in graphic language using the LabVIEW facilities.
- Quick-look software – Telemetry Viewer which deals with all non real time telemetry data. It was also implemented in graphic language using the LabVIEW facilities. An unique configurable program was developed to process in real time and in playback all types of SACI-1 telemetry packets by selecting one of the possible types as relevant telemetry. In playback operation mode, many instances of this process can be run concurrently allowing the Ground Station operator to monitor different types of telemetry packets on the same screen. Despite the detailed data analysis of each experiment data set being done later on at scientific environment, **TMViewer** software system was developed for fast evaluation of these data at Ground Station. These processed data can be stored in files and distributed to the scientists by e-mail giving them early information about the mission operation status related to their experiments. Additionally, **TMViewer** software allows the operator to separate the telemetry raw data received during the entire passage in different files, one per type of telemetry packet: PLASMEX, ORCAS, PHOTEX, MAGNEX, Solar Sensor, Housekeeping, Stored TM and DUMP.
- Automatic storage of the payload raw data. It performs automatic storage of the raw data measured by on-board experiments in CD-units and in Optic Disk units for distribution. These payload telemetry data are delivered to the experiment investigators, according to an operational proceeding. Additionally, the payload telemetry application packets are decompressed and stored in the database as active data[5]. The "very old data" called historical data are transferred to optical disk indexed by a predefined period of time (e.g.: per week). Backups and all other Data Base Management System (DBMS) utilities such as loading and performance monitoring are provided by the database server with an efficient fault tolerance system and roll-forward recovery (Pereira, 1998).
- Remote Control of database providing facilities for queries and reports using Web Page. Through them the scientific community can access the SACI-1 remote sensing data.

Specially for authorized users, the satellite system engineers and experiment investigators DV&DC system provides telecommand query forms (Fig. 4). That means, in order to generate the SACI-1

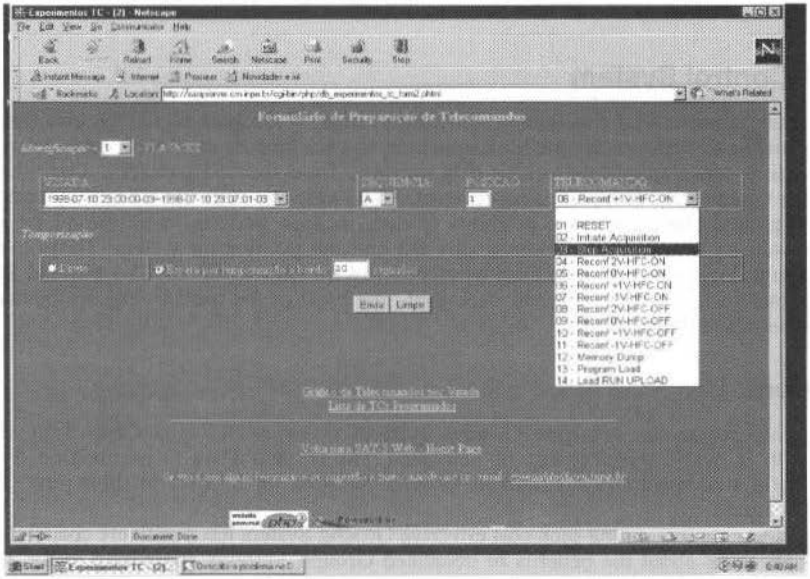


Fig. 4 PLASMEX Telecommand Form

Flight Plan, telecommand messages will be, at first, selected via web pages, allowing users to organize, schedule and edit them focusing on the predicted passages information supplied by the Antenna Control and Positioning System (item 3). With this new approach, authorized users can organize and prepare their telecommand messages, sending it via web pages to the ground station operation. Basically, it is a client-server web-based system compound of some interface pages with dynamic fill-out forms whose command data (telecommands) selected by users are stored and maintained in a relational PostgreSQL database server (Fig. 5).

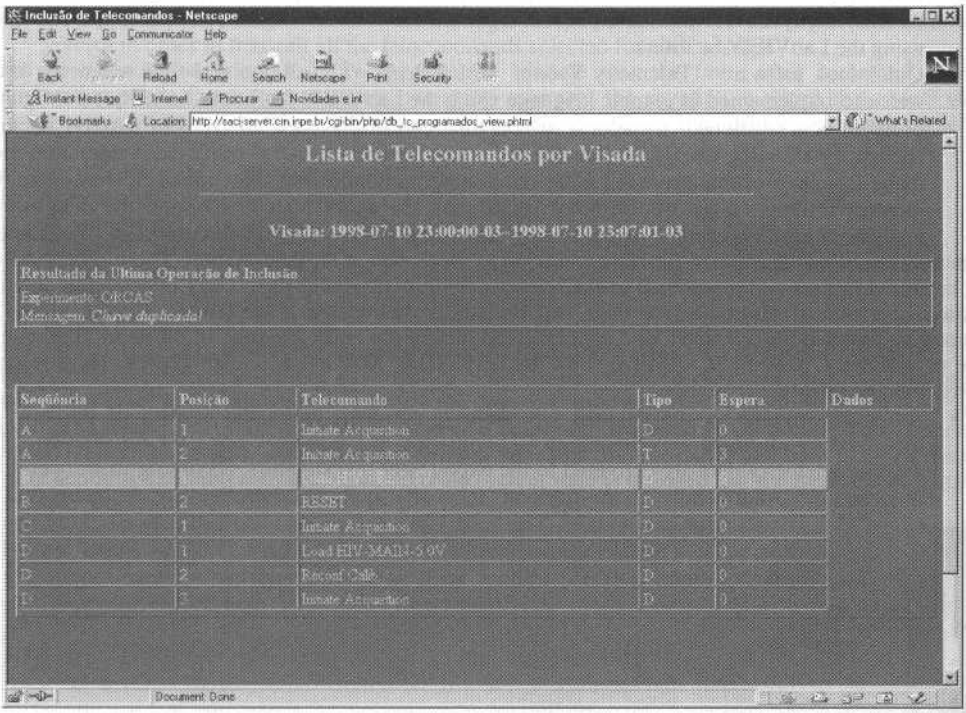


Fig. 5 ORCAS Flight Plan in Data Base

Antenna Control System

The Antenna Control and Positioning software system is the part of the ground station Antenna System responsible to perform the satellite acquisition and tracking. It deals with orbit determination and orbit propagation considering the tracking data obtained from the last passages. The orbit parameters evaluation are based on the Doppler measurements. Moreover, the Antenna Control system provides the satellite and ground station event predictions (passes, eclipse periods, etc.). Based on such events, Antenna software system controls automatically the antenna positioning system. Those events are also available through local network to be used by other software systems of SACI-1 Ground Segment. For instance, events such as, beginning and ending of the passage are used by the Automata program to carry on telecommand transmission.

Conclusion

The use of commercial available Ground Systems (such as Antenna Control System and PCs networks) with in house development of dedicated TM/TC and Data Dissemination Systems made possible to build a compact and autonomous Ground Network for SACI-1 in short time (2 years) with low cost (3 persons).

The experience on integrating such cost effective Ground System has positively contributed to change the way people think about the benefits of dedicated Ground Systems comparing to the multi-mission Centers.

References

Neri, José A. da C. F. et al., September, 1996, "SACI-1 - A Cost-effective Microsatellite Bus for Multiple Mission Payloads". International Conference on Small Satellites: Missions and Technology. Madrid, Spain.

Carvalho, M. J. M., 1997, "Implementação e Testes do Protocolo da Camada de Transferência do Sistema de Telecomando da Estação Solo do Satélite SACI-1". M.Sc. Thesis. PPgEE/UFRN. Natal, Brazil.

Carvalho, Manoel J. M., April, 1998, "Development of a Communication Protocol for a Scientific Applications Satellite". In: Proceedings of the IASTED International Conference: Computer Systems and Applications. Irbid, Jordan, March 30.

Mattiello-Francisco M. F., Guimarães M.S. and Carvalho M.J.M., September, 1998, "An experience in designing software for microsatellite control using graphical language". – 4th International Symposium – Small Satellites Systems and Services, 14-18 – Antibes, Juan Les Pin – France.

Mattiello-Francisco M. F., Neri, José A. da C.F.and Pereira Jr., R. A., 1997, "A Relational Database for the Space Plasma Data Collected by the SACI-1 Mission". In: Proceedings of SPIE Conference – Multispectral Imaging for Terrestrial Applications II. Vol. 3119. SPIE: San Diego.

Pereira Jr., R. A., 1998, "SAT-5: Um Sistema de Informações em Solo para os Experimentos a Bordo do Satélite de Aplicações Científicas – SACI-1". M.Sc. Thesis. COPIN/DSC/UFPB. Campina Grande, Brazil.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 191-193

ISSN 0100-7386
Printed in Brazil

Deep Space Network Ground Communication Services

Richard W. Markley
NASA/Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive M/S 303-404
Pasadena, CA 91109
richard.w.markley@jpl.nasa.gov

Abstract

The U.S. National Aeronautics and Space Administration (NASA) Deep Space Network - or DSN - is an international network of antennas that supports interplanetary spacecraft missions and radio and radar astronomy observations for the exploration of the solar system and the universe. The network also supports selected Earth-orbiting missions. This paper describes the ground communications network of the DSN.

Fundamentally, the DSN ground network architecture is a star network, and the hub is at JPL in Pasadena, California. Communications to customer sites are designed to minimize NASA costs and may be either shared IP backbone networks or dedicated circuits. One of the primary features of the network is its ability to support real-time data, voice, and video communications among antenna stations, an automated multi-mission operations systems facility at JPL (AMMOS), and mission operations centers (MOCs) at NASA and non-NASA facilities.

This paper describes the global ground network, current and emerging requirements, and indicates where new technology is being applied. Major drivers for a new approach are a desire to keep infrastructure costs as low as possible, desire to use the public Internet for communications, wide-scale development and availability of commercial Internet-standard products, need for increased security, and increasing needs for fault tolerance.

Keywords: *Spacecraft, ground network, Internet, communications.*

Background

The DSN has three major antenna sites: 1) Canberra, Australia, 2) Goldstone, California, and 3) Madrid, Spain. Canberra and Madrid are supported with a primary circuit (T1), and a backup circuit (64kbps). The primary circuit is channelized into flight network Internet Protocol (IP) bandwidth, voice and video channels, and administrative IP bandwidth. The antenna station at Goldstone is similar, but has two T1's for data and voice. A separate T1 at Goldstone supports radio astronomy.

The flight IP channels are used for transmitting spacecraft telemetry, command, and tracking data. Station monitor and control data also shares the bandwidth. An upgraded monitor and control system is almost ready to become operational, and this will require additional bandwidth to support the information infrastructure.

The IP bandwidth is sized to accommodate worst-case real-time traffic loads, i.e., occasions when all antennas may simultaneously be active with real-time downlinks from spacecraft.

Another emerging requirement is to provide a common management infrastructure within the DSN based on world wide web technology. Our administrative bandwidth of only 64 kbps is seriously outdated.

JPL also operates a multimission operations facility (AMMOS) for JPL missions. Therefore a significant amount of traffic flows over local area networks (LANs) to that facility. Data to other missions is routed immediately to wide area network (WAN) circuits anywhere in the world. At the present time this includes circuits to domestic U.S., Japan (NASDA), Germany (GSOC), France (CNES), and Canada (CSA).

The Organization

The Network Infrastructure Services (NIS) organization provides both Operations and Engineering. In Operations, NIS has communications technicians who oversea the WANs 24 hours per day, seven days per week (24X7). We also manage a team of system administrators who maintain and sustain over 500 workstations and over 100 network devices, most of which are in the AMMOS facility.

In Engineering, senior system engineers are responsible for 1) Communications and 2) Data Services. The Communications system engineer coordinates customer data, voice, and video requirements with our WAN provider, the NASA Integrated Services Network (NISN). The Data Services system engineer coordinates network monitoring and metrics, data delivery services, and distributed services such as AFS (Andrew File System) and the Distributed Computing Environment (DCE).

The last key team member is the Security Engineer who is responsible for implementing the technology that ensures the security of the flight network. This person manages the firewall systems that separate the IP traffic on the flight network from IP traffic on the Internet.

In the next sections, features of our architecture will be illustrated using the Cassini mission as an example customer.

Circuits

After initial processing at the stations, Cassini downlink data is combined with other flight data from the antenna stations and transmitted to JPL. This real-time IP bandwidth is about 988 kbps from Canberra and Madrid, 1.5 Mbps from Goldstone. It is processed in AMMOS, where mission operations takes place. Huygens probe data is transmitted to a Science Operations Processing Computer (SOPC) in Germany. Science data is transmitted outside the firewall to ten SOPCs at various scientific institutions.

Project personnel may communicate with station personnel over secure voice nets. These are dedicated synchronous channels that carry compressed voice signals.

Station personnel may also participate in NASA activities with a dedicated channel carrying low-bandwidth NASAselect video. Dial-up (ISDN) desktop video conferencing is available in Canberra.

IP Bandwidth

The DSN has a requirement to support an Emergency Control Center (ECC) in Goldstone that is capable of performing the functions now performed at JPL in the event of a natural disaster. For this reason we have ordered a high-capacity backup circuit (768 kbps) that will enable operations from Goldstone to Madrid and Canberra. We have specified this circuit to have a termination at Goldstone and at JPL, so the circuit can be used as backup on a day-to-day basis.

With this design we will use Multi-Link Point-to-Point Protocol (MLPPP) through the primary routers to give an effective IP bandwidth of 1,766 kbps (988+768) to the overseas stations. In the event of failure of one of the circuits, the remaining circuit can carry the traffic.

Telemetry is bursty. Therefore, we plan to interface administrative networks to the same MLPPP router so they can participate in this greatly increased bandwidth when telemetry volume is low. Real-time telemetry data will have priority over administrative data. This will provide a dramatic improvement to administrative bandwidth.

In addition, the NASAselect video will be converted to an IP H.323 standard format, at a relatively low priority, which will free the bandwidth now dedicated to video for use by other applications.

Data Services

JPL has added several data services to the infrastructure that enable reliable (no errors or loss of data) real-time transmission of spacecraft data.

Customers have the option of traditional delivery (now call low-latency data delivery- LLDD) in which real-time data streams from the station in UDP packets with no error corrections.

The major improvement is a customer option for fault-tolerant data delivery - FTDD, in which telemetry data passes from the telemetry processor to a Reliable Network Server (RNS) server at the station. The RNS transmits streams from the station to an RNS server at JPL using TCP, and automatically buffers data and makes corrections if necessary over the potentially noisy satellite circuits. There is a slight latency penalty - 5 to 10 seconds.

The RNS servers were designed to account for all data delivery. Therefore, one of the benefits of this architecture is that all data can now be accounted for on a mission-by mission basis, and provide a basis for fair chargeback whenever and if ever that may be necessary. Until now, we could only measure general IP channel utilization using the metrics associated with the routers.

Mars Global Surveyor, Cassini, DS-1, Mars Climate Orbiter, Mars Polar Lander, and Stardust use the RNS service.

Another data service is a Central Data Recorder at JPL which stores mission data for specified periods of time. The CDR captures both LLDD and FTDD data and has proven useful to recover data when anomalies have occurred.

Security

The flight network is a protected network, and we adhere to NASA security requirements. With increasing numbers of international partners, and the growth of the public Internet, information technology (IT) security has become an extremely important aspect of flight operations.

The flight and administrative LAN networks at the antenna stations are physically separated. At the present time we separate the flight network and administrative network flow between JPL and the stations at the multiplexer (so they share a common WAN). We are completing experiments now with the goal of letting both functions share the MLPP router so both types of data can perform dynamic bandwidth allocation.

Our architecture includes modern IP firewalls at the following two locations:

1) At JPL there is a firewall through which missions can send data from the flight network to users (e.g., scientists) over the public Internet.
2) In addition there is a firewall at Lockheed Martin Astronautics in Denver, CO, which develops and operates NASA's Mars missions.

Network Management

The growing requirement to reduce operations costs has led JPL to centralize network management as much as possible. We have implemented network management software that is capable of collecting statistics, warnings, etc. and forwarding them to a central server located at both the communications area and the system administration area. These tools are proving to significantly reduce our time to isolate problems and restore service.

Our operational IP environment requires personnel with both IP router skills and WAN circuit skills. In JPL's case these are two different groups. To enhance coordination between the teams, JPL has installed an action request system. This system helps document problems and provides a historical log for later analysis.

Conclusion

The architecture of the DSN flight network is evolving to support dynamic bandwidth allocation based on IP. Requirements for administrative traffic and video are two beneficiaries of this development. Also, non-real-time mission data will be delivered more quickly to our customers.

With an architecture based on Internet Protocol (IP) we find wide-scale development and availability of commercial Internet-standard products applicable to our environment. Commercial network management tools have helped us make significant progress in reducing the time to isolate network problems.

Lastly, our use of products based on IP, and the desire of our customers to use the public Internet for transmission, has greatly increased our dependence of the latest firewall technology.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 194

ISSN 0100-7386
Printed in Brazil

Data Processing Systems I

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 195-202

ISSN 0100-7386
Printed in Brazil

Ariane Tracking and Location Real-Time Applications on a Classical IP Network

Nicolas Hugues
Laurent Becquey
Bernard Dellery
Bernard Froment
Michel Studnia
CNES, Sous-Direction Développements Sol
Division Stations et Mesures
18, avenue Edouard Belin 31400
Toulouse Cedex 4 - France

Abstract

In 1992, CNES and ESA decided to renew most of the operational systems of the Guiana Space Centre in Kourou, in order to prepare for the launch of both Ariane 4 and Ariane 5. A new architecture had been designed for the measurement system of the Ariane Launch Complex, and six concurring projects were decided on. Interfaces between all operational systems have been standardised: IP transmission protocol over an extended Ethernet network has been chosen for either inter or intra system data communications, including real-time applications.
This article describes the objectives and real-time constraints of the measurement system of the Ariane launch complex, presents the architecture and main characteristics of each operational system, and highlights the choices which have been decided on for the design of each system in order to successfully complete the new Ariane measurement system with real-time application on an extended IP network.
Keywords:*Real-time application, IP protocol, Ethernet network, tracking, telemetry, location, data processing, operation control.*

Objectives and real-time contrainsts of the Ariane launch complex measurement system

The real-time missions of the Guiana Space Centre during an Ariane launch are mainly:

- to track and locate Ariane from the lift-off to the end of the mission, after satellisation of the payload,
- to receive, store, and process Ariane telemetry data,
- to check with both telemetry and location data that Ariane is nominal and safe for the population,
- to co-ordinate the operations of the whole Ariane launch complex.

From the mission contrainsts, which were listed, the most difficult demands to meet were the reliability and availability requirements for providing real-time information to the operator responsible for the safety of people and goods during the Ariane launch.

These stringent requirements were given as follows:

- to provide at least two different pieces of information about the launcher trajectory and any possible danger for the population, using two different processing subsystems physically separated from end to end;
- to display information within 200 milliseconds after the acquisition of the measurements on ground at Kourou.

Architecture of the measurement system

Four main operational systems are involved in real-time applications and safety information during an Ariane launch.

Figure 1 (below) gives the functional architecture of the real-time applications in the measurement system of the Ariane launch complex, and Fig. 2 (next page) gives an overview of the operational systems involved in real-time applications.

The **Ariane Telemetry System** (SYSTA) receives the telemetry from the launcher, stores it, extracts and transmits the main parameter values to Kourou, where they are processed and displayed in real-time.

The **Location and Trajectory System** (SLT) tracks the launcher with radars, calculates in real-time its location in space, using radar or telemetry data, and displays the trajectory and any potential danger in order to ensure the safety of people and goods.

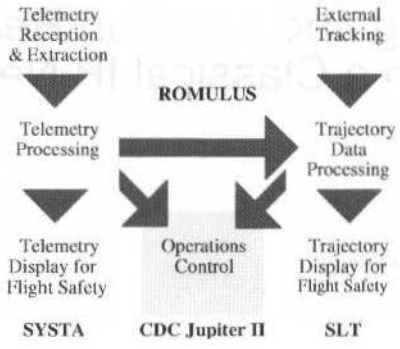


Fig. 1 Functional Architecture

The **Operation Control Centre** (CDC Jupiter II) manages the whole operation at upper level. For this purpose, it centralises information from all the systems of the Ariane launch complex and displays synoptics to each top level manager.

The **Ground Telecommunication Network** (ROMULUS) provides voice, data, and video communications between and within all the systems, including for real-time applications.

System design

Designing the new measurement system, CNES decided to standardise data interfaces and communication services. Access to Ethernet network and IP transmission protocol were imposed for data communications between and within each operational subsystem.

For the particular case of real-time applications, it was therefore decided that the timing of the data exchanges would be managed, if required, at the application level, in order to preserve it from any disturbance implied by the non-determinist Ethernet and IP protocols (collision, repetition, jitter, ...).

In addition, an evaluation of each piece of data to transmit was made for each operational system, in order to specify the network performances to provide and the minimum bandwidth to guarantee for each individual exchange.

Excepting these requirements, each project was free to design the architecture of its operational system, either in choosing the hardware, or in defining the software environment (operating system, COTS software, and specific applications). Choice was mainly dependent upon the functional requirements and the existing equipment of each operational system.

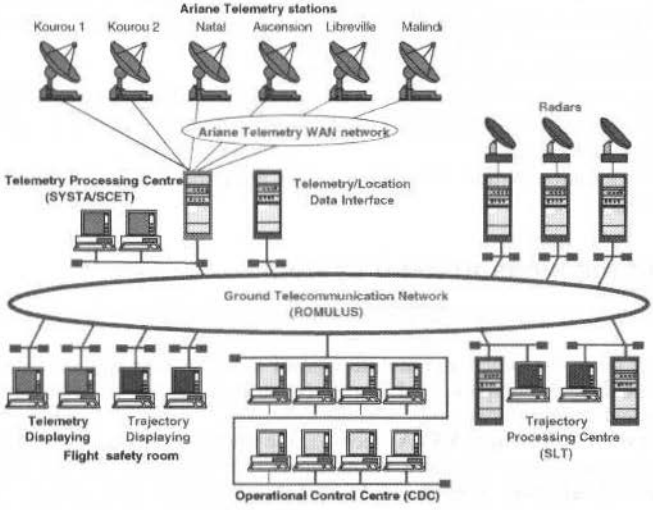


Fig. 2 Overview of the architecture of the Ariane launch complex real-time measure system

The Ariane Telemetry System (SYSTA)

SYSTA general architecture

The SYSTA system consists of

- five S-band Telemetry Stations located at Kourou, Natal in Brazil, Ascension Island in the South Atlantic Ocean, Libreville in Gabon, and Malindi in Kenya.
- a Telemetry Processing Centre located at Kourou (SET/SCET),

The telemetry stations belong to CNES or ESA. They are used for eastern launches (GTO). In case of northern launch (SSO), they are replaced by American or North Canadian stations.

Each Telemetry Station consists of

- one antenna (two in Kourou),
- demodulation and decommutation equipment for acquisition,
- tape recorders for storage,
- a processor for extracting some parameters from the telemetry,
- and communication devices.

The SYSTA System includes a private digital network which interconnects the remote stations to the Telemetry Processing Centre (SET/SCET) in Kourou, using NASCOM or HDLC protocols.

The telemetry processing centre in Kourou includes two redundant subsystems which both process the same telemetry data and display in real-time parameters to operators who analyse the status of the launcher during the flight.

Two remote terminals are located in the flight safety room to display some of these parameters for the Safety Officer.

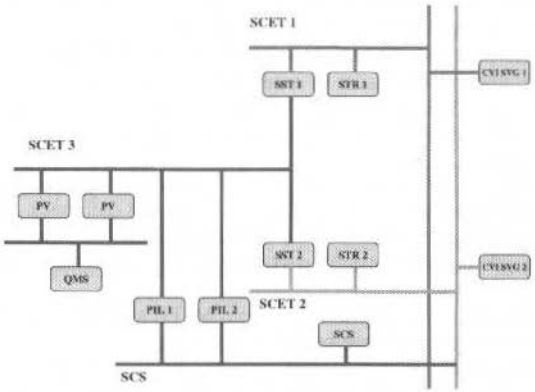


Fig. 3 SYSTA Functional synoptic

SYSTA real-time contrainsts

The main real-time contrainsts for the SYSTA System depend on the telemetry rates to process. For Ariane 5, telemetry stations have :

- to receive and store 1 Mbit/s telemetry for 10 minutes (250 kbit/s for A4),
- to extract in real-time and send to Kourou 38.4 kbit/s telemetry data (9.6 kbit/s for A4).

The onboard location and trajectory extracted from Ariane Telemetry (at Kourou) have to be sent to the location data processing centre in less than 200 ms.

SYSTA design concept

In order to benefit from the TCP/IP quality of service and to meet the required performances, the telemetry processing subsystems use several separate local networks, each one dedicated to a function, some of them having very few terminals connected to them. The following networks have been provided:

- two local networks, one for each of the telemetry data real-time processors (SST) and their real-time backplane (STR)
- a local network for telemetry display (PV) and the system pilot (PIL)
- a local network for printing (QMS)

- a local network for co-ordination and status control (SCS)
- and two remote networks, one for each of the flight safety officer terminals (CVI SVG)

The Location and Trajectory System (SLT)

SLT general architecture

The SLT system consists of

- three S-band Radars, two located at Kourou, and one at Cayenne (60 km),
- two computers which are interfaced to the telemetry data processing centre in order to extrapolate onboard location data from telemetry,
- a data processing centre which receives external location data from the radars and onboard location data.

To achieve its objective, SLT system has been designed to distribute a number of functions. These remote functions exhibit high cohesion within themselves and loose coupling between each other. Individual computers and application software are used to perform these functions. This basic philosophy has shaped the design of the network.

- On each radar site, a rack-mounted computer (EIR) is directly interfaced to the radar in order to successively:
  - capture raw data from one radar (20Hz),
  - transform the data into a co-ordinate frame (at 10Hz),
  - correct it from refraction disturbance,
  - time stamp the data,
  - send it out over the network (at 10 Hz) to the location data processing centre
  - and then transmit launcher location data back to the radar.
- In the telemetry data processing centre, two workstations (MITE) receive onboard location data from the SYSTA telemetry processors in order to successively:
  - capture the data (at 2.5 Hz for A4, at 3.48 Hz for A5),
  - extrapolate the data into a co-ordinate frame (at 10 Hz),
  - time stamp the data,
  - send it out over the network (at 10 Hz) to the location data processing centre
  - and then transmit launcher location and point of contact data back to the telemetry system.
- In the location data processing centre, two workstations (TR1 & 2) receive the same data from the EIR and MITE workstations through two physically separated networks. Both TR1 & 2 successively:
  - capture telemetry and radar location data from MITE and EIR workstations (at 10 Hz),
  - filter radar location data to reduce noise,
  - transform the localisation data into trajectory and location data,
  - evaluate in real-time the potential dangers which could be involved by a failure of the launcher,
  - time stamp the trajectory and safety data,
  - and finally send it out over the network (at 10 Hz) to the display function.
- In the flight safety room, two workstations, RSV and ISV, receive the trajectory and safety data from TR1 and TR2 respectively and display them for the Safety Officer (10 Hz).
- All other workstations which are not used in real-time are connected to a preparation network which is isolated from the real-time traffic during operations.

SLT real-time contrainsts

The location information must be provided to the flight safety officer no longer than 300ms after acquisition by a radar (time measured < 270ms). Time is reduced to 200ms for information from telemetry data (time measured 170 ms).

The information provided to the flight safety officer must be calculated and refreshed at least every 100ms.

The dating accuracy of location data must be better than 100 microseconds (time measured < 50 μs).

For this purpose, each real-time computer has been equipped with an interface board receiving universal time and countdown time IRIG signals, clocked at 1000 Hz, in order to time stamp each piece of data.

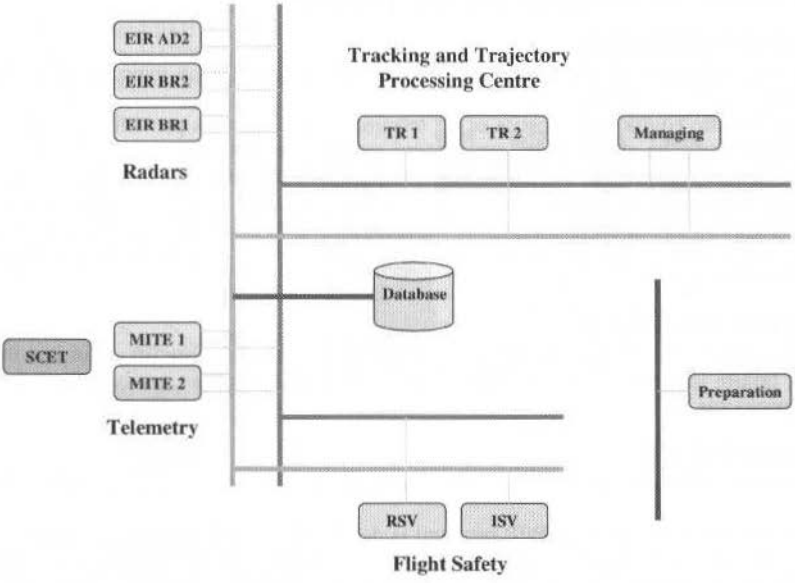


Fig. 4 SLT Functional synoptic

SLT design concept

TCP/IP is a connection-oriented protocol, which provides a reliable full-duplex byte stream for a user process. This protocol is used for all non real-time messages during preparation.

On the contrary, UDP/IP is used for all real-time communications during flight operations, nevertheless this connectionless protocol does not guarantee that the datagrams ever reach their destinations. In fact, due to the very brief lifetime of each piece of data, it has been assumed that the loss of a datagram would not be a problem if the following datagram arrives at its destination in time. And the best solution was, indeed, not to repeat a message but just keep the network free for transmitting the next datagram.

Another constraint was to prevent collision on the network. Since each real-time computer receives a clock at 1000 Hz, it was decided that all the SLT software processes would be divided in 10 Hz cycles, and each cycle subdivided in three phases during which each piece of data would be emitted at a precise time. The phases A (20 ms) and C (10 ms) are used to transmit critical data and the phase B (70 ms) to transmit status and command control data.

The clocking of the data emitted by each computer during the three phases of a 10 Hz cycle is shown on Fig. 5.

MITE1	I	CR	
MITE2	I	CR	
EIRs	LCR		
RSV	CR		
TR1		BR CR	TR
RCR		CR CO	
RCTDL		CR CO MM	
Phase	A	B	C

Fig. 5 SLT Transmit chronogram

I = Location data
CO = Command
BR = Synchronisation broadcast
CR = Status control
TR = Trajectory / designation
MM = Best mean

The Operation Control Centre (CDC JupiterII)

CDC- general architecture

The top-level managers of the operational systems are located in the main room of the **Operation Control Centre** (CDC Jupiter II).

The control centre equipment consists of 35 terminals and 2 servers providing an application which collects and processes the status of all operational systems and displays synoptics dedicated to each operator. The servers also provide operators with computerised operational procedures and timetables.

The status collecting application is interfaced with the data processing centres of the Ariane Telemetry System (SYSTA), the Location and Trajectory System (SLT) and other support systems (optic and video, power and air conditioned generation, weather forecast, ...) in order to get information about the status of the equipment of the systems.

CDC real-time constraints

The most critical time constraint of the CDC is to stop the launch before lift off in less than 200 ms, if an important failure is detected in one of the systems. The goal has been easily achieved using TCP/IP and UDP/IP protocols.

CDC design concept

Two different networks are used to collect and display the data acquired from the operational systems.

The first network which uses TCP/IP protocol is dedicated to external CDC communications for the transmission of status data from the operational systems of the launch complex to the CDC server. A specific socket is used to interconnect each entity and a port number is assigned to each type of data.

The second network is dedicated to internal CDC communications. It uses UDP/IP protocol to broadcast the data processed by the CDC server to all the workstations on which each operator can display the synoptic of the operational systems according to his password defining the pieces of information he may monitor. This network is also used to update the hot redundant CDC server.

The Ground Telecommunication Network (ROMULUS)

Contrainsts

The ROMULUS network provides all communication services with data, voice, video and time information over a wide area.

Data communication services must be provided:

- with a high level of reliability regarding their importance during Ariane launch phases,
- with a high level of performance to respect real-time requirements of telemetry and location systems,
- with a high level of adaptability, evolutivity and standardisation regarding the operational systems to be interconnected now and in the future.

For this purpose, the IP transmission protocol and the 10baseT Ethernet interface have been chosen and imposed on the new operational systems as the unique standard for data communications.

IP network design concept

The IP network must support both real-time and asynchronous communications and guarantee the availability of the services and performances. It has been created with two different parts, one for real-time services, and one for non-real-time services.

Real-time communication services

The compatibility of the IP network to real-time communications has been obtained by the use of very simple rules which guarantee the network performances. The architecture of the real-time communication network is given on Fig. 6.

First, the network architecture was based on a star topology using 2 Mbit/s serial links between routers associated to a mechanism of bandwidth reservation: 80% is reserved to IP protocol, 20% is allocated to others (e.g. SNMP for net-management).

Then, the pieces of network equipment dedicated to the two redundant subsystems of the location data processing centre were doubled, and the links from the remote sites were carefully created in order to guarantee the physical independence of both subsystems from end to end.

Also, direct links interconnect the routers involved in real-time communications. The routes are static and there is no secondary route in case of traffic congestion.

Therefore, the physical link used for each particular exchange is identified anytime and the transfer delay is guaranteed for each UDP/IP packet between routers.

In addition, an Ethernet subnetwork has been dedicated to each functional subsystem.

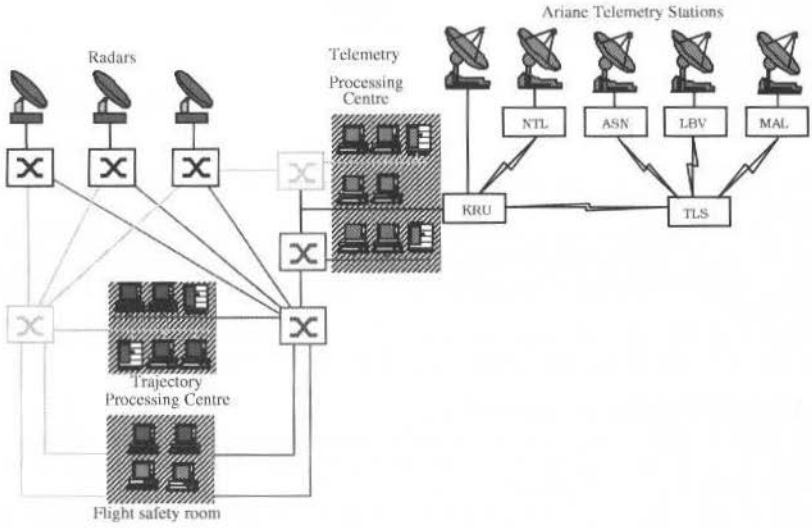


Fig. 6 Real-time communication network architecture

In this architecture, there is no network overload due to the fact that each application controls its own traffic: when required, the timing of emission is clocked at the application level for each piece of data while there is no outgoing traffic emitted by the parties waiting for it.

Network performances

The 2 Mbit/s links used to interconnect the IP routers are created on the ROMULUS transport network, which is stratified into three layers (cf. Fig. 7):

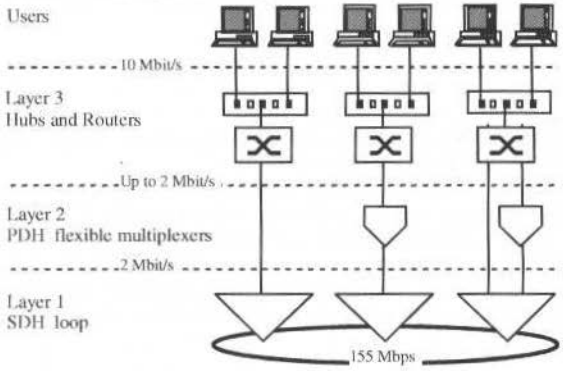


Fig. 7 ROMULUS transport network layers

- The first layer is based on a 155 Mbit/s SDH loop on optical fibres; it provides 2 Mbit/s digital serial links between eight remote sites (up to 63 links).
- The second layer is based on flexible multiplexers and PDH transmission links (2, 8, and 34 Mbit/s) on optical fibres and radiolinks; it provides low and medium rate point-to-point data links up to 2 Mbit/s; it is also used to provide audio point-to-point circuits.
- The third layer is dedicated to Ethernet user access, LAN internetworking, and IP protocol routing. The 2Mbit/s links provided at layers 1 and 2 are used to interconnect the WAN network equipment.

The most remote routers to be interconnected concern the location data processing centre and the radar in Cayenne, 60 km from Kourou, which is linked to the ROMULUS network through a radiolink.

With the architecture detailed above, the maximum transfer delay of a 256-byte data packet between the radar in Cayenne and the location data processing centre in Kourou is:

• Ethernet LAN emission	250 µs
• Router emission	1300 µs
• Transmission on PDH and SDH	825 µs
• Router reception	1300 µs
• Ethernet LAN reception	250 µs
Total	3925 µs

This is compatible with the 5 ms allocated to the network within the 100 ms frame of the SLT real-time application.

However, this transmission delay does not include protocol hazards like time-out (15 ms), additional transmissions (5 ms) or acknowledge delays (5 ms), the sum of which can reach 30 ms. It is assumed that these possible hazards are avoided at the application level by waiting for a clock before sending any data through the network.

After more than one year of operational use, it is now established that, due to the network architecture, these IP protocol hazards, though acceptable, almost never occur.

Non real-time communication services

The Operation Control Centre requires asynchronous communication services. They are mainly dedicated to status data collecting. However, this traffic had to be carefully considered in order to be transparent for real-time communications.

These asynchronous communication services have been provided regarding the following concepts.

Within each operational system, the processing centre of this system collects the statuses of each piece of equipment of each subsystem.

Immediately after a status change, the processing centre forwards the collected information to the status application of the Operation Control Centre.

This TCP/IP data is routed through the network using the OSPF protocol. This part of the network also uses a star topology, but is based on specific 2Mbit/s links independent of the links dedicated to real-time communications.

These asynchronous communications are filtered in order to guarantee performances and security. Due to the star topology, the filters are concentrated into a unique but autoredundant router. Therefore, it simplifies the filtering configuration and has a very small consequence on the performance of the network.

Conclusion

Development of the new operational systems of the Ariane measurement system began in 1994. They were installed from 1995 to 1997. They are now operational.

On December 21, 1997, the 104th Ariane was the first one tracked in real-time with remote distributed applications using IP transmission protocol over a wide area Ethernet network.

Nevertheless, the fact that the architecture of each operational system was designed differently, and after more than one year of operational use, it is now established that due to the global architectural design, the clocking of data emission at the application level, and bandwidth precautionary reservation on the ROMULUS network, real-time communication services can be provided using classical IP routers and simple Ethernet hubs.

Addendum

The new operational systems described in the paper above have been respectively designed and produced for CNES by the following European contractors:

- SYSTA : Dassault Aviation (F), Trasys (B), CIR (CH)
- SLT : Logica (UK), Sema Group (F)
- CDC : GTD (E)
- ROMULUS : MMS (F), Spacebel Informatique (B), Nortel (UK)

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 203-210

ISSN 0100-7386
Printed in Brazil

ENVISAT Monitoring and Control Facility (MCF)

Ismael López
Miguel Belló
Ricardo Moyano
Luis Gonzalo Gutiérrez
GMV S.A
Isaac Newton 11, PTM Tres Cantos, E-28760 Madrid (Spain)
ilopez@gmv.es

Abstract

ENVISAT is a remote sensing satellite to be launched next year, within the frame of the Earth Observation Missions of the European Space Agency (ESA). This paper describes the strategy adopted to elaborate the mission plan activities related to the payload data production and handling, focused in particular on two instruments: ASAR (Advanced Synthetic Aperture Radar) and MERIS (Medium Resolution Image Spectrometer). The result of the involved processing is called the Regional Mission.

The Monitoring and Control Facility (MCF) is embedded in the ENVISAT Payload Data Control Centre (PDCC) located at ESRIN, and is responsible of the planning of the Regional Mission. This facility is being developed by GMV within the frame of the PDS contract♦. The planning is performed according to system orders issued by the users at the USF (User Service Facility) using an abstract of the MCF Mission plan (POP). The Planned Operations Plan (POP) is sent periodically to the USF by the MCF, and it is used by the USF in order to warn the user, in case of potential conflicting requests, allowing the calculation of alternative segments. These system orders are sent via the USCF (User Service Central Facility) to the MCF. The planning process considers the different constraints derived from the use of the instruments, the communication link capabilities, and the system orders themselves, in conjunction with the maximisation of the usage of the resources. This activity works in constant co-ordination with the Flight Operations Segment (FOS), located at ESOC, which is responsible of the planning of the remaining instruments. In consequence, both components centralise the planning of all ENVISAT instruments: the MCF the Regional Mission and the FOS the Global Mission. Therefore, the MCF turns out to be an important facility within the PDS as it is a key interface to the Flight Operations Segment, since it provides to the latter the planning of ASAR and MERIS FR instruments via the PEP (Payload Exploitation Plan), whereas the FOS provides the global planning through the DMOP (Detailed Mission Operations Plan).

Keywords: *Mission Planning, Payload Data Handling, Constraints, Conflict Detection, Conflict Resolution.*

Introduction

The payload of ENVISAT is composed by the following instruments (Pfeiffer, B., 1993): ASAR, MERIS, MIPAS, GOMOS, SCIAMACHY, RA2, MWR, DORIS and AATSR. The data handling of ENVISAT is composed of four tape recorders (current MCF baseline) and three channels (one of them redundant) for X and Ka band downlink, whose transmission capacity is up to 100 Mbps. The tape recorders can record up to 5 Mbps. Therefore, instruments generating data at higher rate than this threshold, must be directly downlinked to a ground station (X band) or via DRS (Ka band). The low rate data can be recorded on-board and played back to a ground station or via DRS when available. This states a basic criteria to distinguish between the Regional Mission and Global Mission. The Global Mission comprises the instruments producing mainly low rate data. These instruments are: MIPAS, GOMOS, SCIAMACHY, RA2, MWR, DORIS, AATSR and the low rate instrument modes of MERIS. The Regional Mission comprises all instrument modes of ASAR (high and low rate) and MERIS FR.

The Global Mission plan is generated by the FOS, based on the Reference Operations Plan (ROP). This ROP is generated by the RGT (ROP Generator Tool) which is also being developed by GMV. The Regional Mission plan is generated by the MCF according to the system orders issued by the users and the Background Regional Mission (BRM), which is a set of system orders (High Rate BRM) and a reference operations plan (ROP) for the ASAR low rate instrument modes (Low Rate BRM), determining the operations to be carried out by the instrument in the gaps resulting between operations requested by the users.

♦ Contract number PO-CO-CSF-GS-2020 with ASPI (Alcatel Space Industries) in the frame of the ESA PDS contract, managed by ESRIN.

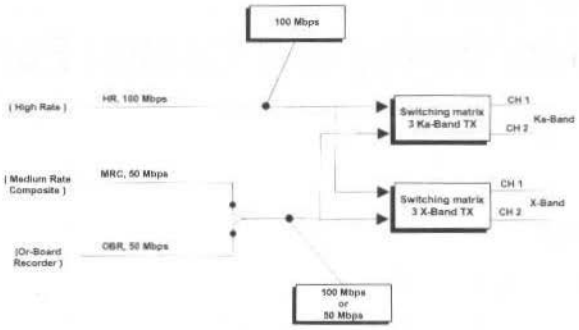


Fig. 1 ENVISAT On-Board Data Handling

MCF General Description

The MCF (Monitoring and Control Facility) is embedded within the ENVISAT Payload Data Control Centre (PDCC) and has associated two major functionality:

- The generation/update of the Regional Mission plan, derived from the processing of the system orders issued by the users.
- The update of the mission plan with the Regional Mission activities validated/rejected by the FOS (Flight Operations Segment) and the Global Mission generated by the FOS.

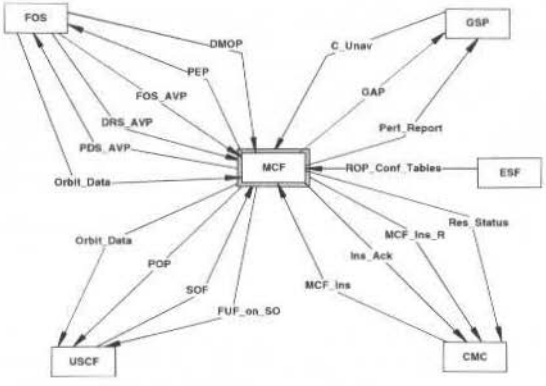


Fig. 2 MCF Context Diagram

As depicted in the previous figure the MCF is not an stand alone system, but interacts with the FOS to co-ordinate the planning activities, and other facilities of the PDS such as USCF (User Service Central Facility), GSP (Ground Segment Planner), ESF (Engineering Support Facility) and CMC (Centre of Monitoring and Control).

The MCF is an operational tool which has been designed following the OMT OO methodology, using C++ as the implementation language.

MCF Overall Planning Cycle

The sequence of events (identified below) to be executed within the MCF define the Planning Cycle.

The USF (User Service Facility) provides the user the capability to issue system order files (SOF). The USF is a world wide web server. There are several USF within the PDS (at least one per PDS centre / station). The system orders are collected in the USCF and forwarded to the MCF. These system orders might be for Regional or Global Mission.

The MCF receives from the GSP the unavailability plan (C_Unav) for the ground segment (i.e. downlink ground stations), and from the FOS the unavailability plan (FOS_AVP and DRS_AVP) for the space segment (i.e. instruments and Data Relay Satellite DRS).

The ESF facility provides the Reference Operations Plan (ROP) tables related to the scenario definition (e.g. reference orbit), the ASAR and MERIS constraint parameters, as well as the ASAR low rate operations plan.

The MCF generates/updates the Regional Mission plan accordingly. Once the system orders have been processed, the MCF generates follow-up files (FUF) on the system orders, which are forwarded to the USCF. This allows the users to keep trace of the status of their requests. The MCF generates the PEP (Payload Exploitation Plan) containing the instrument operations of ASAR and MERIS FR allocated to the corresponding downlink band.

The PEP is forwarded to the FOS which is responsible to validate/reject the Regional Mission. The FOS reports via DMOP (Detailed Mission Operations Plan) to the MCF, the result of the PEP validation, and the overall mission plan during the PEP period.

The MCF generates the GAP (Global Activity Plan) which is forwarded to the GSP. This GAP file contains the activities validated by FOS for both Regional and Global Mission, as well as the system orders issued by the users for Global Mission, allocated to the different stations or archiving centres. For each station, the time at which the downlink transmissions are performed together with the associated instrument data expected during the transmission, as well as the link between the data and the user requests are reported. The GSP processes the received GAP files, preparing the instructions for each ground station or archiving centre.

The MCF generates the POP (Planned Operations Plan) reporting the instrument operations for both Regional and Global Mission, which is forwarded to USF via USCF. This file is used by the USF in order to warn the operator of a potential conflicting request, thus allowing the users to define "a priori" conflict free requests.

The MCF supports both nominal and emergency planning cycles. The activities reported via PEP/DMOP from 2 weeks ahead to the current date, during the period of one week, are known as the nominal planning cycle, whereas all the activities between 2 days and 2 weeks ahead to the current time constitute the emergency planning cycle. Therefore the MCF accepts nominal system orders (issued up to two weeks before the first acquisition) and emergency system orders (issued up to two days before the first acquisition)

Both planning cycles can be configured by the operator, to be executed manually (triggered upon operator requests), or automatically by the MCF (triggered according to regular frequency or upon reception of emergency data).

The Mission Planning System (MPS) is the core of the MCF as it supports its major functionality. For the purpose of this paper we consider more interesting to address the functionality to generate/update the Regional Mission plan. Therefore, from now on, we will focus on the description of such functionality.

MPS Inputs Description

Previously, we have identified three major inputs in order to generate/update the Regional Mission plan:

- System orders of Regional Mission.
- Unavailability plan for the Space and Ground Segment (mainly instruments, downlink ground stations or DRS).
- Reference Operations Plan files.

The system orders for Regional Mission define the observations that the user is interested in. They are set in terms of: acquisition parameters (time of the observation, instrument mode, observation strategy, ...), processing parameters (required processing in order to obtain the product requested), and dissemination parameters (definition of the dissemination of the product to the user). The observation strategies are: zone mapping, containing several segments covering the zone; precise location, containing one segment; conjunction, when several segments of different instrument modes are requested concurrently to observe a zone; and interferometry, which implies several segments of the same instrument mode located in relative close points, overflown in different orbits (ground track deviation of hundred of meters). The user may also define a standing system order for the zone mapping or precise location observation strategy, with a set of number of visiting over the same zone according to a fixed periodicity. Therefore a system order contains all the segments requested by the user, time tagged according to the reference orbit, based in geographical considerations, and "a priori" conflict free. Each individual segment within the system order is called an AAC (Acquisition Atomic Chain).

The unavailability plan is considered in order to reject segments that request an instrument operation, or an acquisition over a ground station which is not available at that time.

The ROP files mainly define: the reference orbit to be used, among others, to time tag all the events and to calculate the ground station visibility; the ASAR and MERIS constraint parameters; and the ASAR

low rate instrument operations which are to be planned during periods when there are not ASAR instrument operations requested by the user.

MPS Planning Loop

The MPS calculates the Regional Mission plan according to: the inputs identified, the satellite and operation constraints, and the optimisation criteria.

The main processing within the MPS is as follows:

1. Perform the priority promotion of the activities already planned, that are close to be carried out (in less than 14 days ahead). As the conflict solving mechanism is based on priorities, the promotion of the closest activities reduces the possibility of their cancellation.
2. Retrieve from the database the segments (AACs) to be processed, grouped by the visiting to which belong. Each group is called an AAC group.
3. Sort the AAC groups by priority.
4. Plan individually each segment of the group. Once all the segments of the group have been planned or rejected, the constraints related to the AAC groups are checked.
5. Identify the gaps between consecutive instrument operations of ASAR and fill them using the ASAR low rate ROP file.
6. Visualise the results of the mission planning session in the Gantt Chart. The commit of the results into the database, or undo the changes, is performed upon operator decision. The generation of the PEP, the POP and the follow up files on system orders, which report to the external facilities the result of the mission planning session is subject to the commit of the database.

MPS Planning Modes

The MPS has two functioning modes (MPS modes): "conflict detection" and "conflict detection and resolution".

In "conflict detection" mode, the MPS checks and informs the operator about any constraint that has been violated, without solving it. The activities which are conflict free are planned automatically. The MPS supports an interactive Gantt Chart to graphically display the remaining conflicts and the planned activities, allowing the operator to manually shift the segments, change priorities, cancel some operations, select alternatives, etc, in order to solve the conflict. This iteration is supported as many times as the operator desires. In this manual iteration, the MPS prevents the commit of a mission plan which is not conflict free.

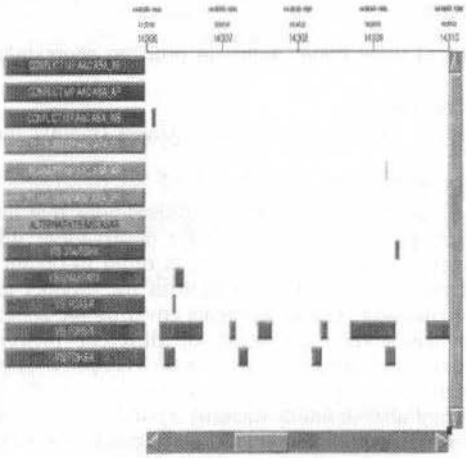


Fig. 3 Conflicts Reported to the Operator

The "conflict detection and resolution" mode implies the automatic resolution by the MPS of all the constraints violated in order to obtain a conflict free mission plan for Regional Mission. All the decisions taken by the MPS in order to solve the conflicts, as well as the conflicts found are reported into a Log file for operator confirmation. This confirmation would imply the commit of the mission plan or the undo of all the modifications carried out during the planning session.

Constraints Considered in the MPS

The MPS deals with different sets of constraints that apply to ASAR and MERIS FR. Basically there are three types of constraints:

- System order constraints, imposed by the user, such as the minimum percentage of segments requested per visiting within the system order.
- Instrument constraints such as:
  - Maximum and minimum duration for an individual instrument operation.
  - Minimum transition time requested between consecutive instrument operations.
  - Concurrent instrument operations of the same instrument in different modes, swath or polar, are not allowed.
- Operational constraints such as:
  - Maximum and minimum usage period for an instrument in high rate modes during a moving window period. This constraint is derived from the duty cycle of the on-board batteries.
  - Merging of consecutive segments requesting the same instrument mode and parameters but with a separation time lower than a threshold.
  - Illumination conditions of MERIS FR instrument operations, at the centre of the acquisition scene.
  - Unavailability of the space or ground segment.
  - Maximum transmission time of ASAR high rate instrument operations on Ka band, within a moving window.
  - Minimum duration of the transmission time of an instrument operation of MERIS FR on X and Ka band.
  - High rate instrument operations limited to the involved ground station/DRS visibility periods.

Mission Plan Storage

The MPS generates/updates up to one year mission plan starting from the current time. The mission plan is stored into a relational database. Thus, a quick access to the data, via SQL queries embedded within the MPS, is supported. Moreover data consistency and protection is provided by the database.

The mission plan consists of five types of events:

1. Ground station / DRS visibility events, identifying the segments where the downlink of data can be performed.
2. Acquisition events indicating when the downlink of data (high rate direct transmission or tape recorders play back) is performed.
3. Instrument operations events, indicating the periods in which instrument are operated.
4. The AAC events indicating the segments requested by the user and its status ("planned" or "cancelled") after the MP session.
5. The unavailability events as derived from the unavailability plan received from other facilities.

All the events keep the links with the associated activities. A visibility event keeps the trace of the acquisition events allocated to it. These acquisition events keep the trace of the instrument operations which generate data that is transmitted during the acquisition events. The instrument events keep the trace of the corresponding AACs that have been the origin of the instrument operation. Thus all the events are linked and the access to the related data is improved in terms of performances.

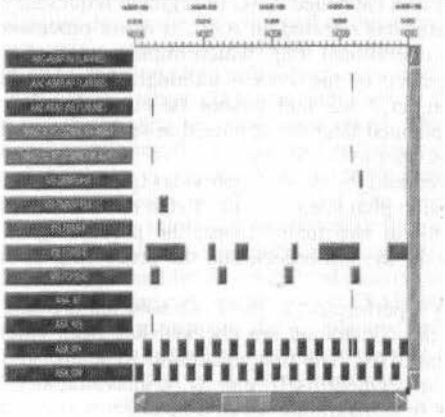


Fig. 4 Mission Plan Representation

All these events can be displayed on the Gantt Chart, as illustrated on Fig. 4, and also on the World Map, as illustrated on Fig. 5. Thus the operator has a graphical view of the mission plan.

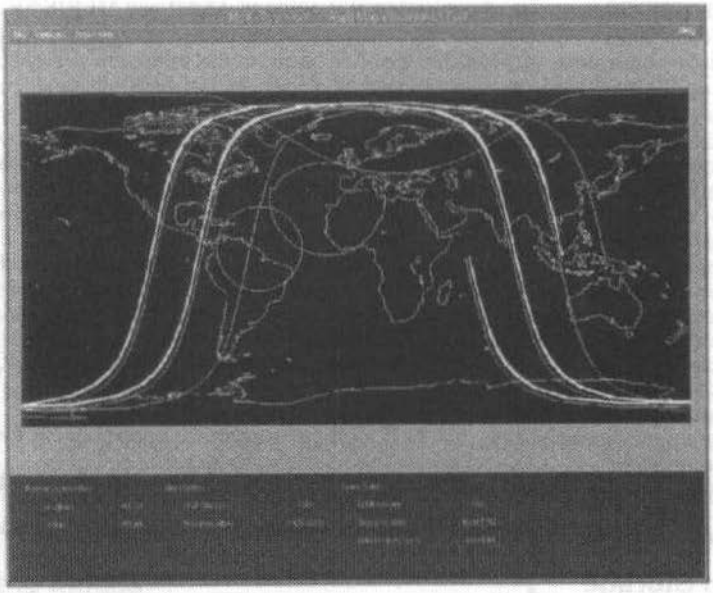


Fig. 5 MPS Alternatives Calculation

Optimisation Versus Performances

The main objective of the MPS is to support the operator to generate, either with manual intervention or automatically, a conflict free mission plan, accounting for the inputs, constraints and the optimisation criteria. The optimisation criteria is stated as: "to plan as much user requests as possible, thus maximising the usage of the resources available for the mission". The MCF performance requirements are also relevant, as it is an operational tool.

These requirements: performances, constraints satisfaction and optimisation criteria are difficult to be fulfilled together, (De Pedro, A., 1995). They constitute the design drivers of the MPS. The way that the constraints are checked and solved, impact strongly in the performances and the optimisation of the achieved solution. Therefore the MPS tries to balance performances/optimisation in order to generate an optimal mission plan, satisfying all constraints, within the performance requirements.

The main aspects considered by the MPS in order to improve the performances are listed below:

1. No further processing of a cancelled AAC (a segment requested by the user) is performed. Once the MPS or the operator has rejected an AAC, it is not processed anymore by the MPS (unless the operator starts an operational loop, which implies a reception of a new system order). This increases the performance of the system although degrades the optimisation of the solution proposed. Notice that an AAC that cannot be currently planned due to the violation of a constraint, could be planned later on in case that such constraints disappear due to an AAC of higher priority coming later to the MCF.
2. The mission plan stored into the database provides quick access to the data.
3. The events in the mission plan keep the links to the related activities.
4. The processing is split in two major loops: the processing of the AAC group, to considers constraints related to the system orders, and the processing of each individual AAC within the AAC group.
5. No further processing is performed, as soon as a solution is found for the AAC group.
6. The order at which the constraints are checked has been carefully designed in the planning process of an individual AAC. The first constraints to be checked are those less time consuming, mainly related with the segment itself (e.g. SZA, unavailability, etc) and the last ones are those more time demanding (e.g. maximum usage time of an instrument within a moving window).

7. The list of conflicts is always sorted by priority, as the conflict solution is driven by the priority of the instrument operations.

The main aspects considered by the MP in order to achieve an optimal solution are listed below:

1. Replanning of alternative segments of an AAC in case of conflicts.
2. Warning the conflicts not solved via replanning but that might be solved by the operator, slightly modifying the AAC (e.g. minimum time for an instrument mode is not satisfied). Then the operator may proceed in "conflict detection" mode to solve them.

Planning of an AAC Group

As stated previously each group of segments (group of AACs) is processed independently. Each AAC of the group is processed individually but keeping always the link to the associated AAC group. At this stage the constraints related to the system order are considered.

Before processing the AAC group a rollback point is marked within the database. Thus, at the end of the processing, if the whole AAC group is "cancelled" the MPS rollbacks to the mark to restore the initial status. Nevertheless the cancellation of the AAC group is reflected in the database.

The groups are processed in three different stages:

1. Firstly each AAC of the group is planned individually, operating the MPS in conflict detection mode. Only the AACs of the AAC group without conflicts are "planned". The constraint of the minimum number of segments requested is checked. In conflict resolution manual mode it is the end of the processing of the AAC group (the remaining conflicts will be reported), as well as in conflict detection and resolution mode in case that previous constraint is satisfied.
2. In a second round, the AACs planned in previous sessions of the MPS with lower priority and in conflict with AACs of the AAC group are replanned, thus allowing the planning of the last ones (see next section).
3. In a third round, if the constraints of the minimum segments requested is not satisfied at previous step, the remaining AACs of the group in conflict are replanned. After the replanning, if the minimum percentage requested is not fulfilled the whole AAC group is "cancelled".

The Mission Planning System Conflict Resolution

This section describes the processing of an individual AAC (a segment requested by the user). It starts identifying, whether the AAC is new or it represents modifications upon an AAC already processed. These modifications could be either cancellation by the user or modifications of some parameters by the operator. In both cases the AAC and all the linked activities (instrument operation and acquisition events) are removed from the mission plan. In case of cancellation the constraint of the minimum number of segments required by the user must be checked, because the cancellation of one AAC could imply the cancellation of the all AACs belonging to the same AAC group if such constraint is not satisfied.

Now in case that the AAC is new or modified (but not cancelled) further processing is required. The MPS marks a rollback point. An instrument operation event is created as derived from the AAC. The MPS checks the ASAR and MERIS FR instrument and operational constraints. The order in which the constraints are checked are:

1. The instrument unavailability during the time interval of the instrument operation.
2. The sun zenith angle constraints, only for MERIS FR instrument operations.
3. The maximum and minimum duration of an individual instrument operation.
4. The selection of the acquisition station. The user indicates in the system order the preferred acquisition stations to which the data must be downlinked. The MPS generates the available visibility for such stations, combining the physical visibility retrieved from the database and the unavailability associated. With the visibility events the MPS generates all the combinations and sorts them according to the preferences of the user. The generation of such combinations consider the constraints such as the minimum or maximum time required for X or Ka band. The first combination covering completely the instrument operation will be selected, and the corresponding acquisition events shall be generated and inserted into the database.
5. The check of non concurrent instrument operations of ASAR instrument. At this stage, all the instrument operations of the same instrument, already planned, within the period of the influence period of instrument operation to be checked, are obtained from the mission plan and sorted by priority. The conflicts may arise due to different instrument modes, or identical instrument modes. In the first case the lower priority instrument operation must be replanned. In the last case the merging of the instrument operations and the linked acquisitions is to be performed.

Before the merging the maximum operation time constraint must be checked (if violated th merging is not performed). At this step the minimum transition time required is also checked.

6. The constraint of the maximum usage period of the high rate instrument modes during a movin window is checked. In case of violation of the constraint, the minimum set of instrumen operations of lower priority to be replanned, necessary to solve the conflict, are identified.

In conflict detection mode, not all the constraints are checked, but as soon as one constraint i violated no more processing for the AAC is performed. In conflict detection and resolution, as soon a one constraint is violated, the affected instrument operations are identified and those with lower priorit are replanned. Notice that this could imply the replanning of activities previously planned that could b cancelled in case that the replanning is not successful.

The replanning of an AAC implies to find alternative segments to the original one, requested by th user, accounting for a time span defined in the system order, and the zone to be observed. One AA could be replaced by one or two alternative segments, that overlap and cover the original segmen requested. These alternatives are automatically calculated by the MPS, using the adjacent orbits to th requested one. Now the alternatives segments identified will be planned by the MPS in a similar way as the original AAC (using the same algorithms). Therefore the replanning of an AAC could imply th replanning of another AAC with lower priority previously planned, whose replanning could again imply the replanning of another AAC of lower priority, and so on, thus leading to the replanning of several AACs of decreasing lower priority of the mission plan. As the MCF is an operational tool, the number of the replanning chained is limited.

If the planning of the alternative segment is not possible, the MPS will cancel the corresponding AAC. Again in this case, the constraint of the minimum number of segments must be checked, cancelling the whole AAC group in case of its violation.

Conclusions

The MCF is an advanced tool to generate and update the Regional Mission plan of ENVISAT. It has been developed with the state of the art in terms of the design methodology (OMT) and implementation (C++, ILOG Views, ORACLE).

Moreover the planning algorithms have been carefully designed and implemented in order to obtain the most efficient solution regarding the performances issues, constraints to be satisfied and optimisation criteria.

It is a powerful planning tool as it supports conflict detection and conflict detection and resolution modes. Thus the operator may participate in the conflict resolution, by interacting with the Gantt Chart where all the displayed conflicts can be manually adjusted.

The MCF has been developed in a short time schedule (15 months) in two phases. The first version (V1) supports all the interfaces with other facilities, and has been successfully validated and delivered, and it is currently being tested and integrated in the PDCC. The second version (V2) supports all the MCF functionality and is to be delivered in March 1999.

Acknowledgements

To the remaining members of the MCF team of GMV: J. A. Tejo, J. Arranz, F. Bertrand, M. Lizondo, A. J. Fernández, A. Monge and A. Gutiérrez, for their dedication and effort devoted to the successful completion of the MCF.

To M. Wybierala from ASPI, for giving continuous and helpful support. To W. Fouquet from ASPI, for dealing with the management aspects of the project. To J. Y. Le Bras for his support. To M. Irle, from ASPI for trusting GMV to handle the MCF.

To F. Martín Crespo, E. Duato, I. Petiteville from ESA (ESRIN) and F. Demond from ESA (ESTEC) for their valuable considerations of the work performed.

To ESA for endorsing the ENVISAT project.

References

Pfeiffer, B., Gardini, B., and Central, J., 1993, "Envisat and the Polar Platform: The Concept and Its History". *ESA Bulletin*, N. 76.

De Pedro, A., López, I., López, E., Galán, J. J., Paganini, M. and Duesmann, B., 1995 "MAEOBS: A Mission Analysis Environment For Earth Observation Missions". *V Workshop of Artificial Intelligence and Knowledge Based Systems For Space*.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 211-218

ISSN 0100-7386
Printed in Brazil

Why Spend Money for the Development of Project Specific M&C Systems?

Christian Knauer
CAM GmbH
D – 82005 Gilching
christian.knauer@cam-comp.de

Abstract

In the area of spacecraft ground control systems, generalized monitor and control systems can be used for both telemetry processing and telecommanding of the spacecraft and monitoring & control of the associated ground system. Additional utilization is possible for checkout (EGSE) and also AIV (assembly, integration and verification) of the S/C. The basic purpose of this kind of application is always the same: monitor data packets are received, checked for correctness (e.g. sync-word, time-out, correct length of data packet, etc.), the information contained within these data packets is processed and the related information is made visible to the operator. On the sending side any (valid) control information is transmitted to the "equipment" (either on operator request or automatically after detection of specified events) and an appropriate reaction of the equipment is expected.
It is clear that different requirements exist for each project, but the baseline is always the same, only expressed in different words. The only significant differences are the additional real project specific requirements but also these can be found in various – not in all – project requirements in a similar form with minor variations.
The paper will show the most important basic– and also some of the specific - requirements in more detail and will show how these requirements can be fulfilled using FRAMTEC (Framework for advanced monitoring, telemetry and control) as a commercial software product. Since FRAMTEC is a proven product, some experiences with regard to the utilization will be shown.
Keywords: *Data Processing, Generation of Control Information, high capabilities of COTS software, FRAMTEC*

General Requirements

The most recent general requirements to be fulfilled for each data processing system are shown in a the logical sequence for both monitoring and control:

Data packets/frames are to be received from the related source and to be checked for correctness, i.e. checks are to be performed for correct synch word (if available), correct length, timing, missing packets/frames, etc..

Afterwards the information within the packet/frame is to be extracted and evaluated. This information – a set of various parameters - is normally stored at appropriate positions/offsets within the data packet/frame in form of binary data with a variable number of bits (1 – 64) or bytes for each parameter. These bits are to be interpreted as signed or unsigned integer, sign and magnitude or also as float/double in various formats (e.g. IEEE or MIL-Std). The resulting value is the called the raw value of the parameter.

The raw value of the parameter is then to be transformed into the appropriate engineering unit, called the calibrated value. Typical methods are interpolation, polynomial and scaling (i.e. special polynomial) for analogous values and table/state code generation for discrete enumerated values. The resultant binary values have then to be transformed, i.e. formatted, into a readable ASCII string and shown to the operator in alphanumerical and/or graphical representation.

In order to increase the safety of operations the calculated / processed values have to be checked against appropriate limit thresholds, i.e. if the value of the parameter is outside a specified range an alarm message has to be generated and the parameter/value is to be flagged as alarm.

Event messages have to be generated not only for transitions between nominal and alarm states of parameter but also e.g. for changing state codes of parameters, reception of invalid data packets, operator input for configuration purposes, etc..

Basic requirements for control are always the same too: (pre)defined control data packets are to be modified / generated and transmitted to the related equipment. (i. e. according to the command to be transmitted the related control information has to be generated and transmitted to the equipment).

A general requirement for both TM processing and TC generation, is that the handling of the various parameters can be described in a so-called process database. The reason is to ease and to speed up any necessary extension and/or modification related to the processing rules of the various parameters and/or data packets/frames.

Last but not least: In most projects a requirement also is to be found suggesting that existing software shall be reused as much as possible. The intention for this requirement is to reduce the risk of development new software and also to reduce the effort for testing.

Specific Requirements

Besides these already mentioned general basic requirements in most projects exist also more specific and advanced requirements to be shown and demonstrated in the next paragraphs. These requirements are demonstrated for both telemetry processing and telecommand generation

TM Processing

It must be differenciated between the handling of data packets itself, i.e. gathering the relation between the data packet and the parameters (called decommutation), and the processing rules of the appropriate parameters.

Decommutation shall be possible not only for fixed allocation within a data packet but also for both multiplexed allocations, i.e. the relation between the data within the frame and the parameter allocation depends on the value of another parameter, and dynamic allocation of data within a data packet, i.e. for ASCII coded data packets the begin of a parameter information has to be "searched" in dependency of other ASCII coded contents in this frame/data packet.

Newer requirements refer also to the depacketizing of (CCSDS) transferframes. This means that these transferframes contain various source data packets to be extracted from the transferframe according to the CCSDS (and also according to the ESA) standards.

Requirements to perform a Reed Solomon decoding on the received transfer frame have also been found. The checking for the correct CRC seems currently to be no longer a specific but a general requirement.

In order to perform appropriate pre-launch tests and also rehearsals of critical maneuvers it is required in many projects to perform these tests on a given scenario time offset configurable by the operator.

The requirement to make available the time stamps of each parameter (reception time), also in dependency of the occurrence/offset of a parameter within a data packet, i.e. summary of reception time of data packet and offset time of parameter within this packet, can be seen together with the above mentioned requirement (scenario times) to be fully flexible in the handling of these "time stamps".

Other requirements reflect to the limit checking functionality: additionally to the other checks (e.g. range) the trend of parameter value shall be checked according to various statistical functions, accessing the history of the related parameter values.

Derived Parameters and Special Processing

Many advanced requirements can be found for the handling of special processing and the handling of derived parameters (parameter where value and state depends on value and state of any other parameters).

Previously the calculation of these special/derived parameters were coded directly into the application itself and did lead to problems well known to everybody.

In modern projects requirements are found, that the handling of special processing/derived parameters shall also be part of the process database.

This kind of requirements means, that there must exist something like an operations language used for definition of special processing/handling of derived parameters.

Command Generation

Besides the general handling of control, i.e. transmission of a predefined or modified control packet, also the construction of command packets with a variable number of arguments is required for space craft commanding; this includes also the construction of related headers according to the actual standards (CCSDS / ESA).

Functionality must be provided to translate command arguments, given in engineering units, into the appropriate binary form – insertion of hex-values by the operator is no longer state of the art.

Additional functionality must be provided for:

- Argument check: correctness of arguments given by the operator is to be checked.
- Pre-transmission Validation: it is to be checked, if the selected command with the given arguments is allowed to be transmitted in the current state of the equipment.

- Pre-execution verification (for time-tagged commands): It is to be checked, if command is allowed in current state of the equipment.
- Command execution verification: the correct execution is to be monitored, i.e. correct acceptance at ground station and space craft and correct behavior of the equipment must be shown to the operator.

The sense of these requirements is to increase the safety of the equipment/space craft to be operated; they can be found in nearly all actual projects.

Please note that this list of advanced requirements is only a subset and can be extended. But also these mentioned requirements demonstrate the increasing complexity of modern projects.

New Development

These and also further advanced requirements can be realized by developing a completely new software. The advantage of this approach is, that all the given requirements can be fulfilled exactly in the specified manner, but this implies also to re-develop software normally already available with minor variations.

Even if various software packages (e.g. from previous projects) can be reused, the effort and also the risk using this approach is extremely high in comparison of using a commercial product.

Effort and Risks

In the past for the establishment of new projects often a complete new software was implemented, i.e. the basic functionality has been developed often with nearly the same results and disadvantages:

- High costs for development
- Much effort for testing the system
- High risk of new development
- Short time for commissioning the system since completely new software projects are seldom delivered on time

Commercial Software

Another (better) approach is the usage of a commercial software package and configuring the project according the related (also specific) requirements - often without any software modification.

Since even the "newest" requirements in this area are already integrated into such a commercial system, normally only the project specific configuration has to be performed – the effort and costs for development and testing and therefore also the risk is reduced to a minimum.

It is clear that not all of the given requirements can be fulfilled exactly in the required manner, but in general the required functionality is already available.

The advantages of this approach are clear:

- Projects are configured: this saves time and money
- Mostly without software modification: i.e. no development risk
- Using of already intensive tested features: You will get a proven system

Example FRAMTEC

In the following paragraphs the advantages and supported features of a commercial product will be shown. As an example the product FRAMTEC has been selected, since this product has been used for multiple missions and projects as shown in the following tables.

Table 1 Control Center Applications

Project	Location
MIR HK	DLR-GSOC, Germany
MOMS-2P	DLR-GSOC Germany
Logic Box	
NAV	
PAF/HK	
PAF/NAV	
NAHUEL I	Buenos Aires, Argentina
EQUATOR_S	DLR-GSOC, Germany

Project	Location
EUTELSAT W1	DLR-GSOC Germany
LEOP	
CHAMP	DLR-GSOC Germany
XSAR	DLR-GSOC Germany
EUMETSAT-Central Facility	EUMETSAT, Germany
ABRIXAS	GSOC, Germany

Table 2 M&C Implementations

Project	Location
NAHUEL Ground Station & Control Centre	Buenos Aires, Argentina
Ku-Band Ground Station	Weilheim, Germany
CHAMP EGSE	DLR-GSOC, Germany DJO, Jena, Germany Launch Site, Russia
EUMETSAT PGS	Usingen,Germany

The name FRAMTEC stands for **FR**amework for **A**dvanced **M**onitor, **Te**lemetry and **C**ontrol. FRAMTEC is a powerful product family to process telemetry and monitor data as well as for generation and transmission of control / command information. Due to the generic approach FRAMTEC is suitable for nearly all Monitor and Control applications not only in the area of space crafts and ground stations.

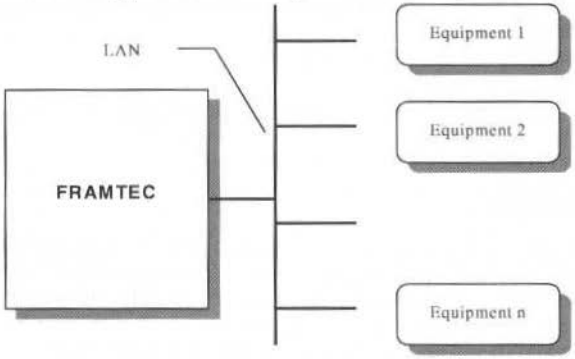


Fig. 1 FRAMTEC Environment

FRAMTEC has generally the ability to process input data which are routed to the system in form of packets (data blocks). The structure of such packets may be of fixed or even dynamic structure. Conversely FRAMTEC generates output packets of any structure with fixed or variable content.

Due to the open structure FRAMTEC can be configured also for multiple space crafts / multiple equipment. The communication is typically based on TCP/IP protocol but also other communication is possible (on request).

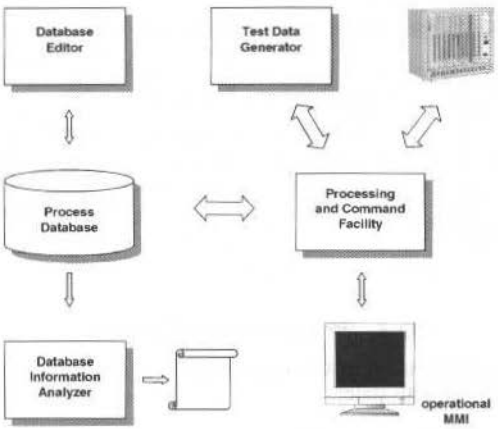
Structure of FRAMTEC

The following figure shows the various components of FRAMTEC in a functional representation.

The FRAMTEC process DataBase FDB contains all the necessary information about the processing of monitoring data as well as the information necessary to generate control data / commands. The process database is loaded into the processing facility during the startup phase.

The component FRAMTEC DataBase Editor (FDBE) is used to establish and maintain the process database(s) (FDB) in an easy and comfortable manner - existing database files can also be reloaded into the database editor for modification.

A **D**atabase **I**nformation **ANA**lyzer *(DIANA)* is provided to check the consistency of a modified database; the verbosity, i.e. the depth of check, can be selected by the user.



PROCOM is the on-line kernel of FRAMTEC – received monitor data packets are identified and processed according to the description in the process database, i.e. the related parameter information - originally a sequence of bits - is translated into a readable form converted into engineering units (volts, temperatures, etc.) and made visible to the operator using the integrated MMI. Alarm and warning generation, event detection, etc. is part of this functionality.

Inputs from the operator are processed and the appropriate action – again according to the database - is performed; this also includes the modification/construction of control information and the transmission of this information to the equipment to be controlled.

During the start-up phase the process database is accessed and the related data structures are built up. After that the facility is ready to perform the appropriate activities:

- Interfacing with the operator, i.e. receiving inputs, execution of related functionalities and generation/presentation of „system" messages, etc.
- Reception and identification of incoming TLM/MON packets
- Processing the received and identified data packets according the description within the process database and generation of the related output
- Generation / modification and transmission of control information on operator request or also on detection of specific events

The MMI (in one or more instances) is a modern Windows based graphical user interface supporting the presentation of the parameter values in various ways. Alphanumerical as well as graphical representation including line plots, symbol plots, i.e. showing graphical items in different colors according the current state are supported.

In many projects appropriate M&C systems have to be configured without direct connection to the related hardware (not yet available or otherwise in use), i.e. there exist only limited possibilities to perform tests with the original equipment. For this reason a test-data generator DATGEN is part of FRAMTEC. DATGEN simulates the appropriate facilities, i.e. it generates related monitor / telemetry information in the same manner as defined for the original equipment and it also receives related control information which can be displayed for verification purposes.

The use of DATGEN reduces the time necessary for integration and test with the original equipment to a minimum.

Features

In the following paragraphs the features of the product FRAMTEC are demonstrated. It is also shown, that also advanced requirements can be fulfilled with this powerful tool. As above in demonstration of the requirements also here the features are shown for both monitoring and control.

Monitoring

In the monitoring part we have to differ between the processing rules of a single parameter and the processing rules of received data packet

Processing rules of parameter

The following figure shows the general processing philosophy of a parameter.

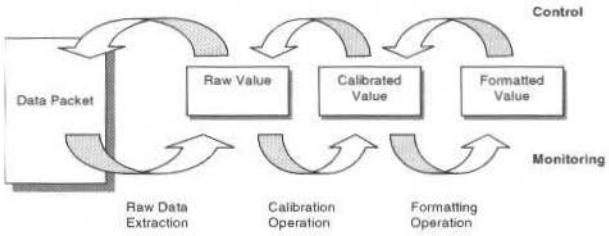


Fig. 2 General Processing Scheme

Various functions can be used to gain the appropriate raw, calibrated and formatted value of a parameter together with the appropriate state:

- Raw data extraction
  - Signed and unsigned integer up to 64 bits,
  - sign and magnitude,
  - float and double in various representations (e.g. IEEE or MIL-Std),
  - ASCII coded values
  - Special processing (see below)
- Transfomration into engineering unit (calibration)
  - Polynomial and scaling
  - Interpolation
  - Table / enumeration
  - Special processing (see below)
- Formatting
  - All commonly used formatting rules (incl. Binary) are available
- Limit and validity checking
  - Range limits with lower/upper and soft/hard limits
  - Trend limits residing on numerous statistical functions referring to the
  - History of parameter values
  - Statecode limits on discrete values (warning and alarm)
  - Adaptive limits, i.e. the currently selected limit set - the appropriate thresholds, depends on values/states of other parameters; this feature can also be used for an automated limit switching on/off.
- Not applicable (N/A) and not selected (N/S) checking
- Validity handling (Not Applicable (N/A), Not Selected (N/S), EXPired (EXP))
- Status consistency checking
- On the other hand the decommutation is to be specified, i.e. in which frames on which positions which parameters are to be processed. Also for this, the decommutation, a wide range of functionality is supported by FRAMTEC:
- Processing of
  - fixed and
  - multiplexed positions within a received data packet;
- Dynamic processing of ASCII coded packets
- Definition and processing of subframes to avoid multiple definition of same parts of data packets
- Processing also information/parameters residing in the header of the data packet
- Timeout check (expired) on packet level
- Memory management/handling
- State vector maintenance to allow access to the current values of parameters from other applications
- Adaptive processing, i.e. in dependeny of value/state of other parameters different sequences of processing rules are executed
- Parameter structures / dependent processing ease the definition of the decommutation
- Lists of derived parameters can be used to ease the definition of the decommutation

Interpreter Language

One of the most important benefits of FRAMTEC is the integrated interpreter language, very similar to 'C/C++', but without pointers. This feature is used

- for the definition of special processing,
- to handle derived parameters within the monitoring part,
- to specifiy conditions for the adaptive processing and
- to define related conditions for various checks in the commanding part.

The appropriate source code is part of the process database, compiled during the database load operation and executed during run-time as virtual machine.

All commonly used language constructs, such as

- global functions and variables
- structures and arrays
- even classes and objects

are available, i.e. nearly the complete C/C++ standard except pointers is supported. The reason for not supporting pointers is that (inexperienced) users could destroy internal data structures by wrong usage of this "feature".

Supporting Features

Beside the mentioned features for the processing also a wide range of supporting features increase the usage of FRAMTEC (only some features are discussed in the next paragraphs)

The Selective parameter page can be used for test purposes. A number of selectable parameters is shown on one page together with the binary raw value, the binary calibrated value, the formatted string and the related flags and stati.

Debug pages are useful to check the implementation of derived parameters. A selected parameter is shown (the current value and state) together with all referenced "input parameters", also with the current value and state; dynamic updates are taken into account.

The automatic configuration check is a useful feature to check once (on operator request) or in regular intervals the consistency and ranges of parameters values of defined parameters. The definition of the parameters to be checked and the related ranges / stati is predefined and read in from file. Any mismatch is monitored and also logged to show the operator the result of the latest check activity.

LADT dump (Latest Available Data Table) performs a print of the values and states of all defined parameters grouped by the related subsystem and sorted in alphabetical order

The Raw data page shows the contents of the last received data packet – of all packets or only for a selected packet – in a hexadecimal manner. This is a helpful feature to investigate any anomalies within the system.

The Limit Page and the parameter information page show the operator for a selected parameter the currently defined limits, including the currently active limit set. On the parameter information page the current definition of the processing rules, including textual description, is shown to the operator.

Commanding

For commanding two levels of functionality are available:

- Simple Control
- Advanced Control

Using the simple control feature , predefined data packets – defined within the process database – can be modified by inverse processing of related control parameters and afterwards transmitted to the equipment.

Inverse processing of control parameters means, that internally the appropriate inverse processing functions are executed in reverse order; no difference is made to normal base parameters for defining them except, that these parameters are to be flagged as control parameters to enable the modification requested by the operator.

The Simple control feature is mostly used for the control of the ground station.

The higher level, i.e. advanced control is much more powerful and normally be used for space craft commanding. This is due to the integrated advanced features such as

- construction of commend packets with a variable number of arguments
- Argument Check, i.e. the validity of given arguments and argument values is checked before further processing of the commend

- Pre-transmission validation (PTV), i.e. before sending the command it is checked that the equipment to be commanded is currently able and allowed to proceed the command to be transmitted. For this, the value and state of related telemetry parameters can be checked
- Command execution verification (CEV), i.e. the current execution state of the command is checked and monitored. Possible checks are
  - Correct acceptance at ground station, i.e. an ACK is received by GS
  - Correct acceptance at the spacecraft; also an appropriate ACK is received
  - Correct execution of command by monitoring the values of related telemetry parameters which have to change their values within a defined time period

All these checks can be defined individually on a per command basis using the integrated interpreter functionality; related supporting intrinsic functions are available.

For the access to values of the referenced telemetry parameters the so called state vector is used. The state vector is implemented as shared memory with functional access:

- The telemetry processing system updates the related values and
- The command system reads out the current values of the parameter; synchronization mechanisms has been taken into account.

This mechanism – the state vector – can also be used by other applications for accessing current values of related telemetry parameters.

Due to the flexible design of this module it is also possible to transfer other information to further applications such as related database information of relevant parameters.

Summary

As shown in the last paragraphs a commercial product, in this case the product FRAMTEC, is suitable to fulfill not only a subset of the requirements in modern projects but is able also to cover all / most of the advanced requirements. Due to the modular structure of this product it is easy to add user-written functions to adapt also project specific functionality.

The question "Why spend money for the development of project specific M&C Systems?" can be answered clearly:

Don't waste time and money for the risky development of new software – use a commercial product and configure your project. Reach the target with less risk in a shorter time and save money.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 219-227

ISSN 0100-7386
Printed in Brazil

Open Center: A Generic Ground System Designed and Developed in a Multi Application Approach

Jean-Jacques Montiel
MATRA MARCONI SPACE, France
31, avenue des Cosmonautes, 31402 Toulouse Cedex 4, France,
jean-jacques.montiel@tls.mms.fr

SCOPE

In 1997, MATRA MARCONI SPACE (MMS) started the development of OPEN CENTER, a generic system for EGSE, Satellite Control Centres, and Simulators monitoring & control.

One of the challenges of this development lies in the fact that it has been on purpose designed and developed in parallel with real application projects, in order to make sure that the resulting product will be perfectly suited to the different targeted applications.

After having briefly recalled the purpose and the contents of OPEN CENTER, this paper details the development process that has been applied to develop a generic product in parallel with utilisation projects. Then, in order to complete the demonstration, the concrete applications of OPEN CENTER to EGSE's (for SPOT5, HELIOS2, METOP, ROSETTA and EUROSTAR 3000), to Satellite Control Centre (for INTELSAT FDC project), and to Simulators (for MSG-SSF and STENTOR programs) are described from a technical point of view.

Open Center Overview

OPEN CENTER provides a standard framework and a consistent set of generic software components that can be used to build EGSE, Mission Control Systems, and Simulators (for monitoring & control purposes). OPEN CENTER aims at maximising reuse from programs to programs, between satellite integration and control facilities, and finally minimising the risk for the customer, in terms of cost and schedule.

Heritage

As shown Fig. 1, OPEN CENTER takes benefits from more than 15 years of MMS experience in ground systems, for

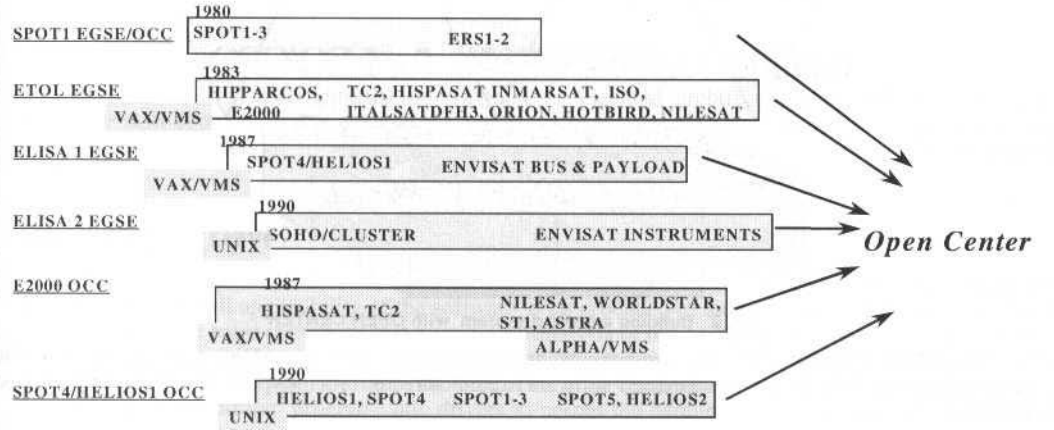


Fig. 1 OPEN CENTER is based on a long-standing experience.

- Satellites check-out Systems (up to 80 EGSE's delivered, 50% of which to external customers)
- Satellite Control Centres (about 12 SCC's delivered)
- Simulators (23 Simulators and Software Validation Facilities for 17 different programs)

Furthermore , a deep analysis of the previous Ground Systems product lines and a study of the existing software components has been made in order to reuse them as far as possible. And finally, the OPEN CENTER general concepts take into account the results of all MMS R&D technological activities in the Ground System domains.

From the early 80th and until now, a number of EGSE and SCC products lines have been developed and intensively used by MMS and external customers for earth observation, telecommunication and scientific satellites.

Objectives

OPEN CENTER proposes the software solution for:

- all ground data systems for Space systems, including test facilities, simulators, control centres, mission centres or integrated ground facilities,
- all type of satellites platforms and applications (Telecom, Earth Observation, ...),
- all orbits (GEOs, LEOs, constellations),
- all phases, from satellite development and testing to satellites operations.
- OPEN CENTER includes:
- A set of generic components covering in a highly modular way the various functions of a Ground System. These components are in addition designed with built-in parameterisation capabilities allowing to adapt them to various contexts.
- A standard framework allowing to host either OPEN CENTER components, or project specific components, or existing commercial of the shelf software components.

The figure 2 gives a representation of one of the main driver of the OPEN CENTRE product policy.

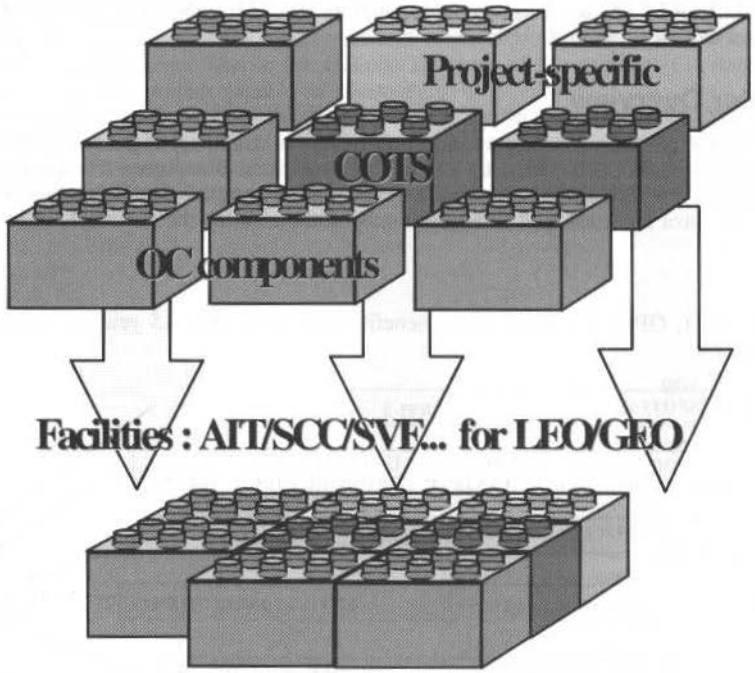


Fig. 2 Building a ground system with OPEN CENTER

The OPEN CENTRE generic components are customised and assembled with project specific components and COTS on a standardised software architecture in order to provide the targeted ground facilities

Content

The Figure 3 details the various components of OPEN CENTER, the utilisation of which in various real application contexts will be shown in the next paragraphs.

This Figure also presents the OPSWARE tools used to support the Satellite operations when required. The OPSWARE tools are designed and developed to be easily plugged on any type of ground facilities.

The main software architectural concepts applied on OPEN CENTER are :

- Independence from the hardware constructors.
- Mono or multi-satellites and mono or multi-machines capabilities
- High availability and redundancy mechanisms
- High life time duration and evolutivity capabilities
- Object oriented design, use of C++ and CORBA
- Optimal use of COTS (ORBIX, ILOG Views, PV-Waves)

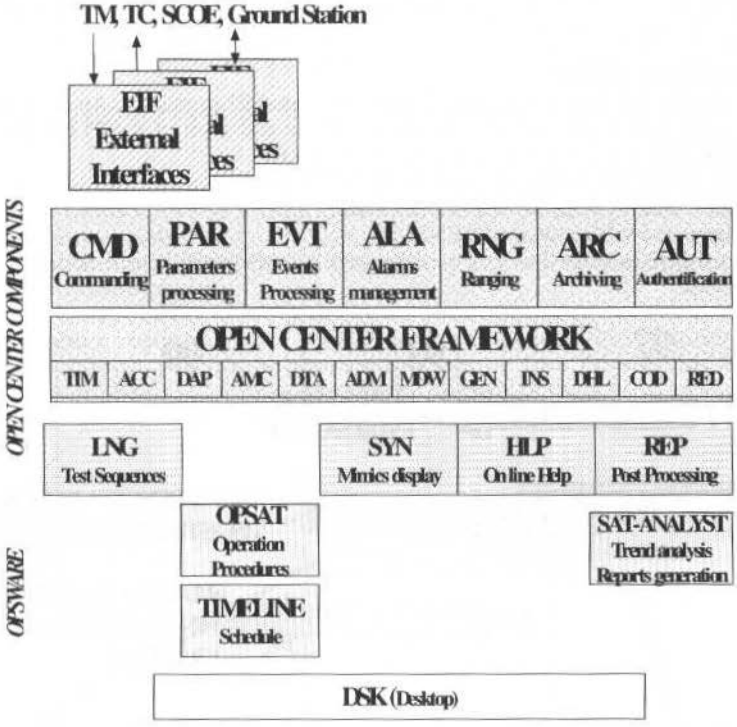


Fig. 3 OPEN CENTER components

The OPEN CENTRE components are structured on a layered architecture with the External Interface layer, the skill components layer (generic and highly configurable to various missions), the framework (communication, timing,..) and the operational layer including the MMI components (LNG,SYN,HLP, DSK, ...)

Open Center Development Process

When MMS decided to start the development of a full generic system, one of the major difficulties was to ensure the full applicability of the software to be produced to the targeted systems and, in order to reach that goal, we were convinced that a very deep analysis of the development process was mandatory.

Thus, the first priority was to establish a complete process model involving the OPEN CENTER Project Group together with the User Projects. The main challenge was to define an organisation, a functioning mode and the programmatic drivers to permit the parallel development of OPEN CENTER components with the user application systems.

One of the main results of the process analysis was to split the OPEN CENTER life cycle in 2 different phases, the construction phase and the routine phase:

- During **the construction phase**, all the OPEN CENTER architectural concepts are progressively set up and the software components are developed to be directly used on the dedicated User

Projects. The construction phase is foreseen for a 2 years duration period (from mid 97 to mid 99) and lead to the availability of a consistent set of OPEN CENTER components based on the same software architecture and covering all functions defined in the OPEN CENTER scope of applicability.

- **The routine phase** (after mid 99) is the concrete result of the MMS product policy in the frame of Ground Systems. It corresponds to the normal operational use of OPEN CENTER components for the Ground Systems Projects. The OPEN CENTER components are still in evolution but on a mode different from the construction phase. The customer projects use the OPEN CENTER components according to their own needs following a complete OPEN CENTRE descriptive and user documentation.

The activities to be performed and the defined functioning modes during these 2 phases are different and both are presented here under.

The Construction Phase

The construction has been defined for 2 years, from mid 97 to mid 99. An incremental development approach is applied during the whole construction phase and a set of 4 OPEN CENTER versions are produced and delivered to the following application projects:

- EGSE's for SPOT5, ROSETTA and EUROSTAR 3000
- Satellite Control Center for INTELSAT FDC
- Simulators for MSG SSF and STENTOR

The Figure 4 gives a general overview of the main programmatic aspects during the OPEN CENTRE construction phase.

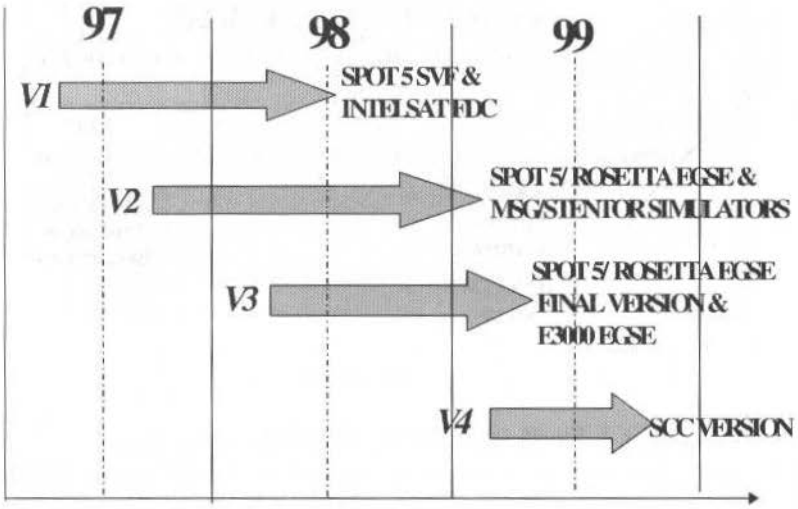


Fig. 4 OPEN CENTRE Releases during construction phase

The construction phase of OPEN CENTRE is fully driven by the User Projects and each identified version is directly delivered for the system integration and validation phase of the targeted projects

For the functional aspects, the operational principle described here after, permits to control all along the construction phase the consistency between on one hand the OPEN CENTER specification process and on the other hand the applicability on the on-going and future User Projects

An **OPEN CENTER User Group** has been set up to be in charge of the OPEN CENTER components User Requirements specifications. This group is composed with a number of MMS Ground Systems end user representatives (AIT/AIV team, Satellite Operations and Simulation teams) . The role of the User group is a major one during the construction phase. It has to produce the full user requirements for all OPEN CENTER components in order to ensure to target the requirements completeness of OPEN CENTER. Then, the User Group has also the mission to validate the User Requirements Specifications towards the User Projects.

The **User Projects** are in charge of the design and the development of the project specific software components. In that respect, they are using and validating the OPEN CENTER external interfaces and the Open Center configuration capabilities for project customisation purposes. Furthermore, the detailed

programmatic of OPEN CENTER is fully driven by the User Projects needs. A versions plan is established at general system level and also detailed at component level according to the development schedules of the targeted User Projects.

The **Engineering Team of OPEN CENTER Project Group** is interfacing the Users Group and the User Projects and provides the OPEN CENTER design and development team with all the relevant information. The Engineering Team performs a very strong co-ordination between OPEN CENTER Project Group and the Users Projects with the objective to set up and to maintain the **OPEN CENTER components Releases Plan** which is the main guideline during the OPEN CENTER construction phase.

On the architectural design aspects, a similar approach involving the User Projects and the OPEN CENTER Project Group is applied. The **OPEN CENTER System Architecture Team**, defining the architectural concepts, is interfacing directly with the different system architects of the User Projects. The consistency of the produced system architecture is verified at User Projects level during the design phase of the to be developed systems. As for the requirement specifications, a lot of efforts is made for the co-ordination of the architectural design of OPEN CENTER and the designs of application systems designs.

The Figure 5 gives a representation of the process model of OPEN CENTRE during the construction phase.

For each identified version, the OPEN CENTER project group delivers to the User Projects an integrated and validated OPEN CENTER version customised for the dedicated project. Then, each User Project is in charge of its own system integration with the specific to project components and at the end of the full system validation. The OPEN CENTER project group supports the User Projects during these system integration and validation phases.

The development process of OPEN CENTER during the construction phase permits us to develop the OPEN CENTER components nearly in parallel with the application systems. This approach, linked to a strong incremental development principle to minimise the risks on User Projects, has been actually efficient to achieve our objectives. Furthermore, the technical diversity of the targeted applications really ensures, to a maximum extent, the OPEN CENTRE generic capability and flexibility.

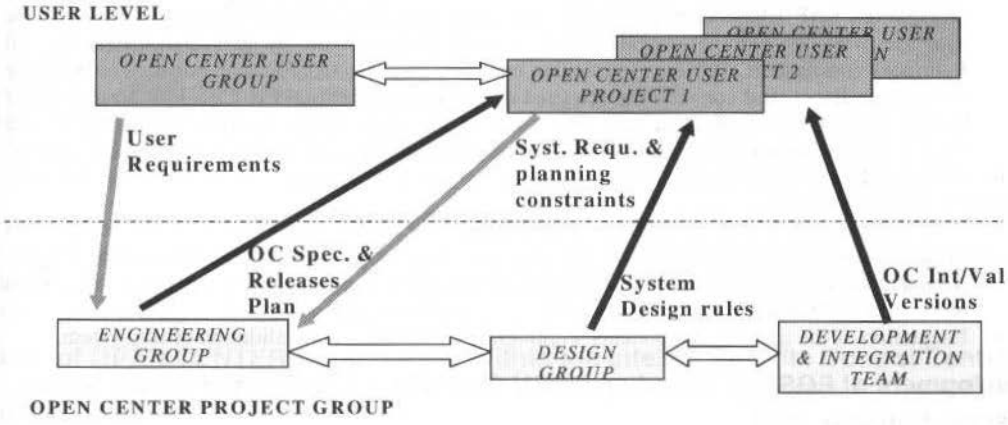


Fig. 5 OPEN CENTRE organisation and functioning principle during construction phase

The OPEN CENTRE specification, design and development activities are driven in strong co-ordination between the User Group, the User Projects and the OPEN CENTRE Project Group

The Routine Phase

The OPEN CENTER routine phase is characterised by the modification of the relationship between the OPEN CENTER Project Group and the User Projects. OPEN CENTER is at that time considered as a provider of software products (as normal COTS) and the User Projects can build their dedicated Ground Systems through a catalogue of the OPEN CENTER components.

The OPEN CENTER project is still in charge of development of new releases and establishes the **OPEN CENTRE RELEASES PLAN** covering the following main activities:

- The **corrective maintenance** of the delivered releases
- The **evolutive maintenance** for the targeted systems new requirements

- The **adaptive maintenance** for the requirements of the new systems to be built
- The **perfective maintenance** for system architecture improvements and follow up of technical evolution

A number of system activities are also performed during the OPEN CENTER routine phase:

- Operational support to client projects and end users using assistance and training means
- Support to system engineering teams for answering to bid proposals, for designing system architectures compatible with the OPEN CENTER components and concepts or even for the definition of necessary adaptations and evolutions
- Setting up and maintenance of an OPEN CENTER reference Ground System facility for OPEN CENTER demonstration and evaluation purposes

Using Open Center in Different Ground System Applications

General Development Approach

The following development approach has to be followed to customise OPEN CENTER for the various application programmes:

- Analysis of requirements, and identification of :
  - The functions that can be implemented using OPEN CENTER components. If the component fully answers the requirements, it can be used as is. Sometimes however, the component had to be upgraded to fulfil the specific requirements of the project.
  - The functions that are specific and require the development of a new component

This analysis is formalised through a compliance matrix that identifies for each requirement the corresponding OPEN CENTER component with the eventual adaptations, or the specific component to be developed.

- Development of the specific components. Most of the time, specific components have to be developed to interface with project-specific input/output equipment. As shown paragraph 2, one important feature of OPEN CENTER is the clear separation between CMD (Commands processing), PAR (Parameters processing including extraction, calibration, monitoring), and the interface with the external world, which is taken into account by dedicated EIF (External Interface) components. For each particular interface, a dedicated EIF component manages the interface protocol, and converts the acquired data into a standard OPEN CENTER format called OCDS (OPEN CENTER Data Set). For most of the projects, the specific developments are limited to the development of EIF components if not available in the OPEN CENTER EIF library.

In addition, OPEN CENTER offers a generic structure ("process skeleton") which allows developing very easily new components.

- Configuration and installation of the reused OPEN CENTER components. Some components include parameterisation capabilities: the functions they are performing can be adapted through parameters that have to be defined within the database. One important part of the configuration work consists then in defining these parameters in the database.
- Integration of the specific components within OPEN CENTER, and validation of the system.

Development of EGSE'S Based on OPEN CENTER : Examples of Spot5, Metop, Rosetta, Eurostar 3000

Generally speaking, the following OPEN CENTER components are used to build an EGSE (see also Fig. 3):

- Common EGSE/SCC components :
  - FRAMEWORK (except redundancy management),
  - PAR (parameters processing), CMD (Commands processing), ARC (Archiving), EVT (Events management), ALA (Alarms management)
  - User Interface components : SYN (Mimics display), HLP (On Line help), REP (Post processing)
- EGSE specific components : mainly LNG (test sequences preparation and execution)

The following specific developments are performed for SPOT5, METOP, ROSETTA, and EUROSTAR 3000:

- Telemetry and Telecommand: on all these programs, there is project specific on the TM/TC interface: classical line/format structure on SPOT5; CCSDS on METOP, ROSETTA, and

EUROSTAR 3000. This is taken into account by specific EIF components, without modification on the PAR and CMD components.

In addition, ROSETTA is using the PUS. This has also led to develop specific MMI to display the packet contents.

- Specific EIF components for interfacing with dedicated input/output equipment :
  - On SPOT5 and METOP, the following specific interface equipment are taken into account : OBA (On Board Access, interface with the On Board Computer for the On Board Software debugging), OBDH on board bus interface, interface with the simulation processor
  - On ROSETTA, a specific protocol is developed to interface with the instrument EGSE's (TM distribution and reception of TC requests)
  - On EUROSTAR 3000, a specific EIF component is developed to interface with the Avionics SCOE (Specific Check Out Equipment).

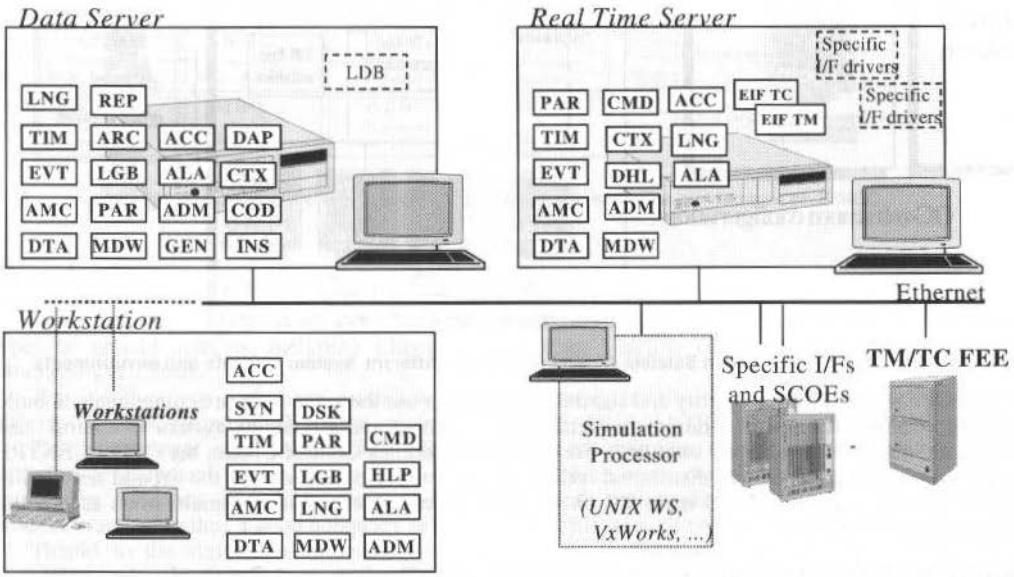


Fig. 6 EGSE typical architecture using OPEN CENTER

The OPEN CENTRE components are distributed on the EGSE Servers and Work Stations. They are linked and integrated with specific software components mainly in charge in this context of special interfaces handling for Simulation means, Front End or Specific CheckOut Equipment.

Use of OPEN CENTER Components Within the Intelsat Satellite Control Centre

The INTELSAT FDC (Flight Dynamics and Commanding) is a good demonstration of the modularity of OPEN CENTER.

As indicated by the name FDC, this project deals with the development of the Flight Dynamics and of the Commanding part of the control centre, excluding the Telemetry that was already developed by INTELSAT.

The Flight Dynamics is based on the MMS QUARTZ++ tool.

The Commanding relies on two main parts:

- The MMS OPSWARE satellite operation support tools, OPSAT and TIMELINE
- A subset of the OPEN CENTER components, mainly :
  - The OPEN CENTER framework, which has been adapted to interface a commercial network monitoring package (Netexpert) instead of using the NMC component.
  - The commanding, CMD component, which has been customised to interface the INTELSAT commanding library.

These components are interfaced with the existing telemetry processing and with the existing database. The CMD and PAR components being independent, interfacing another TM package can be achieved without major problem. OPEN CENTER can be interfaced with any database through its standard ASCII interface.

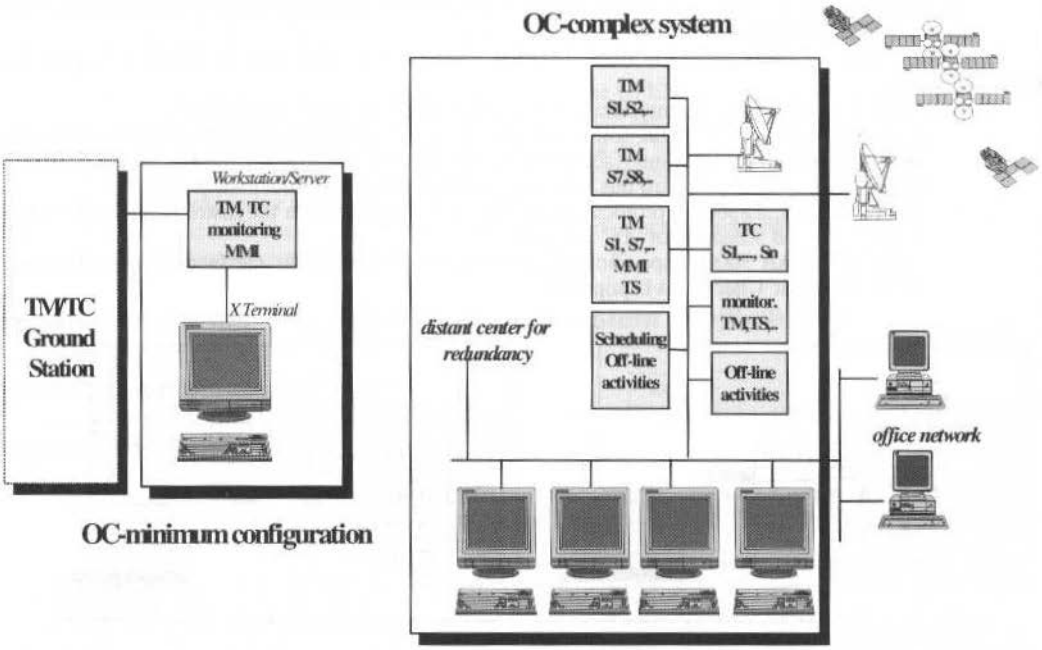


Fig. 7 Use of OPEN CENTRE in Satellite Control Centers in different system contexts and environments

The OPEN CENTRE modularity and flexibility permits to use the same software components to build a Satellite Control Centre in a minimum configuration context and a complex system to control and monitor an important number of satellites. For the multi-Satellites Control Centre, the OPEN CENTRE components are customised and distributed on the Servers and Workstations of the ground network in order to provide a multi-TM and multi-TC chains capabilities together with the multi-users access and redundancy mechanism.

Stentor and MSG-SSF Satellites Simulators Monitoring and Control

OPEN CENTER is used to develop the STENTOR and MSG-SSF Satellite Dynamic Simulators (DSS) monitoring and control (so-called SIMGO).

The simulation functions (development and execution of simulation models) are covered by the MMS SIMWORK and SIMIX components.

SIMGO is based on the following OPEN CENTER components:

- FRAMEWORK
- PAR (parameters processing), CMD (Commands processing), EVT (Events management), ALA (Alarms management), SYN (Mimics display), LNG (test sequences preparation and execution)

Components covering specific simulator requirements have also been developed. Mainly:

- KMC (Kernel Monitoring and Control) : this component monitors the simulation models execution
- FMI (Failures Manager Interface) for the selection and execution of failures within the simulation models
- SCP (Simulation Configuration Preparation)
- SCE (Scenario Editor)
- SAI (Stateset Analysis Interface)

Dedicated component are also available for the Archiving and the post processing, the corresponding simulators requirements being much more simple than for the EGSE and Control Centres.

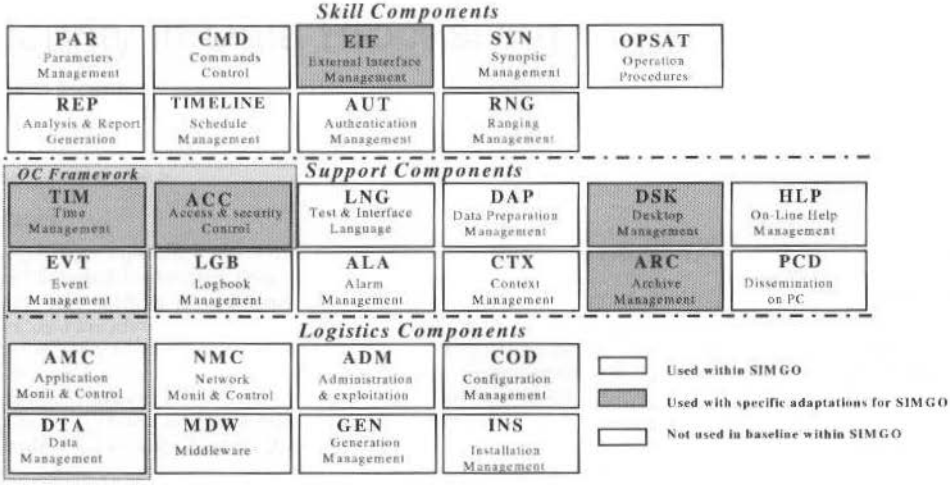


Fig. 8 OPEN CENTRE components used in the software architecture of SIMGO

Conclusion

From the early beginning, OPEN CENTER has been specified and designed to suit to the various types of ground systems, including Check Out facilities, Satellite Control Centres, and Simulators Monitoring & Control.

The specifications have been established with the strong involvement of the internal users of the different systems, taking benefit of the existing experience within MATRA MARCONI SPACE.

The development of the OPEN CENTER components has then been on purpose performed in parallel with the development of the first utilisation projects, which were covering all the targeted fields of application. This has allowed a permanent feedback with real applications during the development process, warranting thus a good adequacy of OPEN CENTER with all types of ground systems.

Thanks to the application to real projects in different domains (EGSE, Satellite Control Centre, Simulator), it has been demonstrated that OPEN CENTER covers all the corresponding requirements and can be easily adapted to various contexts.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp.228-236

ISSN 0100-7386
Printed in Brazil

Security in Data Processing Centre

Richard Moreno
CNES DTS/MPI/PS/TD
18 avenue Edouard Belin
31401 Toulouse Cedex France
richard.moreno@cnes.fr

Abstract

This paper is an experience report from two projects which have been conducted at the French Space Agency (CNES : Centre National d'Etudes Spatiales).
The first one, ScaRaB, is an instrument designed to measure the Earth Radiation Budget from a satellite. In this project, data are provided to the scientific community by the way of Internet.
The second one, EURIDIS, demonstrates the possibility of improving the GPS system by the way of a supplementary satellite for the use of civil aviation. In this project, data are exchanged with external centres through Internet.
The common characteristics of these two projects are that they require stringent security requirements. The objective of this paper is to present how these requirements have been taken into account.
Keywords: *Unix security, Internet, ground segment*

Introduction

Internet and network are more and more often used in ground segments. They both are very powerful tools, but they induce some supplementary risks. In order to avoid these risks, some security principles have to be applied.

The risk is that an intruder can get unauthorised access rights on the system connected to the outside and can:

- read confidential information,
- write or destroy data (Trojan horse, virus, or modification of the content),
- deny normal use of the resources by consuming all CPU, RAM, or bandwidth

The intruder can not only affect the computer connected to the outside, but also the internal network on which it is connected.

It should be pointed out that a security policy induces non-negligible costs and delays. But there is a real need, and a compromise has to be done between the level of security (the price) and the required protection level.

This paper is an experience report from two projects, which have been conducted at CNES: ScaRaB and EURIDIS.

In the first part, ScaRaB ground segment will be presented: what is ScaRaB, and how security has been taken into account. The same topics about EURIDIS will be detailed in the second part

ScaRaB

Radiation Budget

The Earth's climate is governed from the outside by solar energy, which itself depends on both the power radiated by the Sun and the Earth's position in relation to the Sun. As it is at a very high temperature, around 6000 °K (degrees Kelvin), most of its energy is radiated as short waves (from 0.4 to 0.7 mm), in the visible and near infrared (from 0.7 to 4 mm).

In a "simplified" model, incident radiation (visible and near infrared) is absorbed and reflected by the atmosphere. These exchanges and transformations of energy involve a multitude of physical, chemical and biological processes in which oceans, ice, soils and vegetation all play a part.

The Earth's Radiation Budget (ERB) is the net radiation flux, in other words the difference between the solar radiation absorbed (by the atmosphere or land), and the infrared radiation that escapes from the atmosphere into space. The budget for such radiation fluxes, which constitute the only exchanges of energy between the Earth and space, is an essential element in climatic balance.

Climatic change (linked to human activity, for example) can only occur in conjunction with a change in these budgets.

Observing the Earth's Radiation Budget will thus help understand its role in maintaining the Earth's present climate, governing the size and development of large-scale climatic anomalies, and determining how the climate reacts or could react to disturbances, whether their origins be natural (such as volcanic activity) or human (increase in greenhouse gases or aerosols).

ScaRaB project

The Scanner for Radiation Budget - ScaRaB - is as scientific experiment designed to measure the Earth's Radiation Budget from a satellite on which the instrument is flown. This project, part of the World Climate Research Programme, helps further knowledge of our environment and climatic change.

ScaRaB is a cooperative project between France, Russia and Germany. The first instrument model flew in 1994-1995 on the METEOR satellite, and a second model was launched on a RESURS satellite on July 10th 1998, both of which are Russian.

The ScaRaB ground segment has to handle many varied information flows (technological telemetry, scientific telemetry, processing results, telecommand sequences and navigation data) as well as housekeeping messages between the various parts of which it is made.

The level 1a, 1b, 2 and 3 data are processed at CNES. They are distributed to the scientific community by the mean of CD-ROM and through a WWW server. This one provides information about The ScaRaB mission, and also provides means to select and retrieve data from ScaRaB mission.

ScaRaB architecture

The ScaRaB Ground Segment revolves around 2 centres:

- the Russian centre which receives Telemetry and sends Telecommands,
- the Toulouse Space Centre in France (CNES) which monitors and programmes the instrument, and processes, archives and disseminates data

The ScaRaB data products are distributed on CD-ROM and also by the way of a WWW server. The architecture of this server is quite complicated (see fig. 1):

- - an Oracle server contains the list of the data product files.
- - a Storagetek server contains the data products files
- - two operators, who answer to questions, monitor the server and the ground segment
- - a CD-ROM recorder which is directly operated by the operators

The security of such a system is a very strong requirement. Actually, the Oracle database has to be protected. Moreover, the archive on the Storagetek server is very critical.

The telemetry data are received in Moscow, and transferred to CNES in France through a dedicated line which is also used for other experiments (Interball, manned flight, recoverable capsules etc...). Then, these are processed in CNES in order to produce level 1, 2 and 3 data. These data are then archived in a Storagetek, and information about these processed data is stored in an Oracle database. The operators produce CD-ROM containing these data and send them to the scientists.

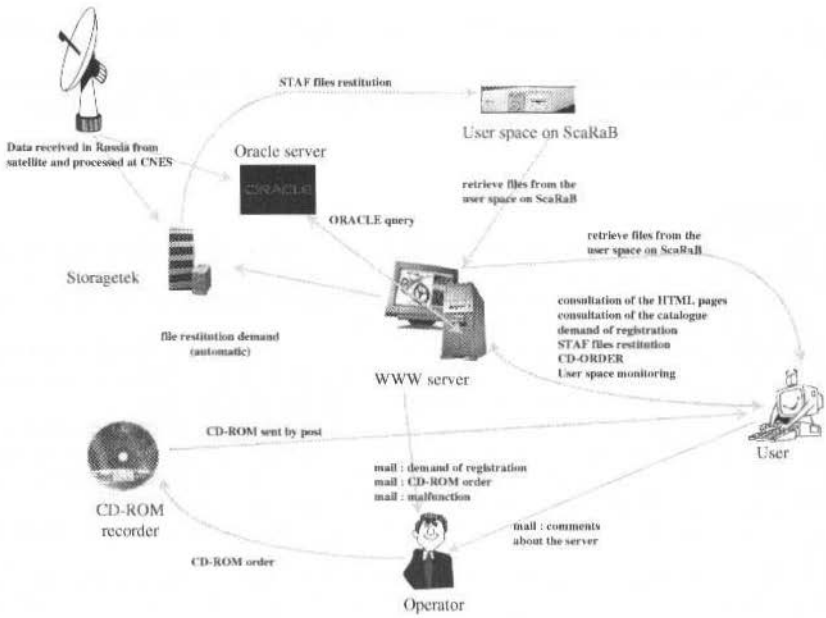


Fig. 1 ScaRaB architecture

A WWW server [1] has been developed by CNES in order to provide tools to the scientific community. With this server, the scientists can interrogate the Oracle database and make query on ScaRaB products. They can also order existing CD-ROM.

The scientists can make query on instrument type, files type, date of data, and cycle of data. Then a files list is presented to the scientists, who can get supplementary information about each selected file or can ask for the ancestry (*files used to create this file*) of each selected file. The WWW server makes these queries to the Oracle server. The scientists can also download files from the Storagetek server. The WWW server forwards the request, and the files are archived on the server. Authorised scientists have tools to download these files from the WWW server.

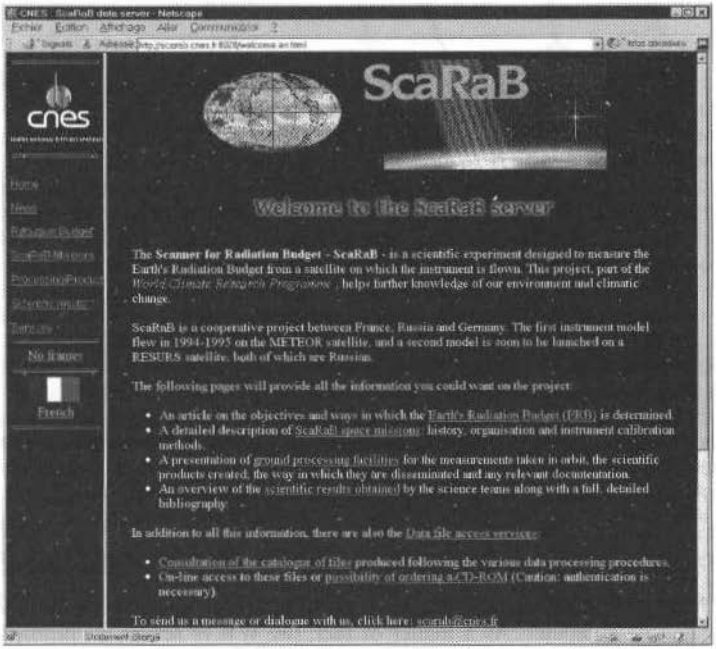


Fig. 2 ScaRaB WWW server

ScaRaB Security

Introduction

Different kinds of security aspects have to be considered. The most obvious one is physical security. The server is stored in a restricted area in order to avoid theft and vandalism. Also a copy of the server is stored in another area whether the disk is destroyed (fire, water...).

It is also very important for the operators and the administrators of such system to have enough skill. So they must keep being formed in order to follow technique and systems evolutions.

But the more complicated security is logical security.

Usual WWW Server Security

A WWW server is only a program which is executed on a computer. This programme (then the computer) communicates with the outside. So communications between the computer and the outside should be controlled, and the security of the computer has to be reinforced. It is also necessary to make sure that the WWW server cannot access to (so does not have the ability to corrupt) the core of the computer.

Figure 3 shows that a router (*device which forwards traffic between networks*) filters the traffic between the CNES network and the Internet. This router has been configured in order to allow only the necessary traffic flow (connection to the WWW server on the WWW server specific port). So, the Oracle server and the archive on Storagetek cannot communicate with Internet, and the computer which hosts the WWW server, can only be reached on one port.

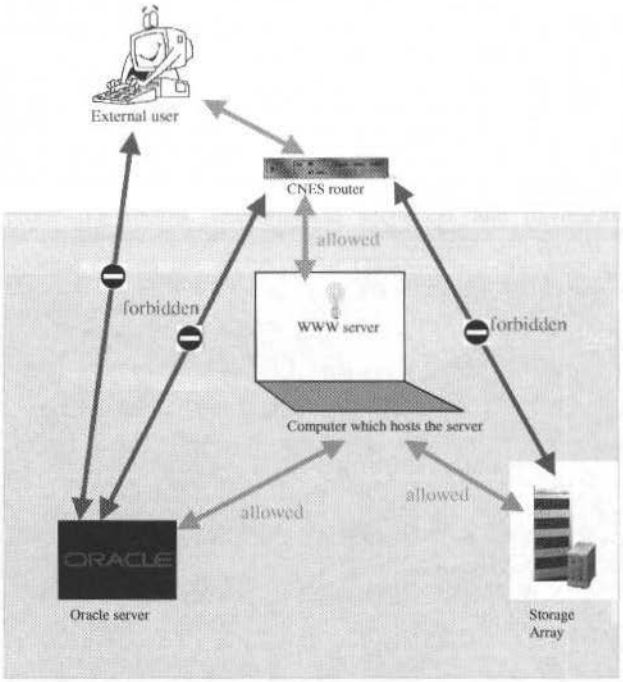


Fig. 3 ScaRaB security

The computer, which hosts the WWW server, communicates directly with Internet. So its security level must be very high. The configuration of this computer is consistent with CNES strongest security standards: permissions mode of system files and directories audit, accounting, ...

Moreover, in order to avoid any attack through the WWW server, a separation has been done between the WWW server and its computer. As a matter of fact, the WWW server is chrooted (*chrooting is a method of protecting part of a disk from potential attacks. When a program runs while chrooted to a particular directory, it sees only files below but not above that directory*) in a second environment which has been rebuilt.

The security of the WWW server is of course essential. A well known server has been used : the sources of the server are available and have been verified, and the configuration of this server is easy and well known. Every access, and also every error are recorded in the server logs. The administrator of the server shall check these logs regularly for suspicious activity

The ScaRaB users have been divided in three groups which have different privileges on the data and the programs. ACL (*Access Control List*) entries have been used in order to restrict access to some files and cgi-bin which are protected so that the remote user has to provide a name and password in order to get access. Moreover these files and programs are protected in such a way that only browsers connecting from certain Internet address can access them.

The tripwire program daily scans the server files and detects if any files or executables have been modified.

Security of Dedicated Software

The locally developed software for this WWW server requires a particular attention. As a matter of fact, any cgi scripts may contain bugs, and every bug is a potential security hole. For example, bugs may allow intruders to execute commands on the machine that hosts the WWW server.

Precautions have been taken during the writing of the source code. Programmers can reduce the chance that a potential intruder can exploit a hole or bug (e.g.: a current security hole is not to verify character buffers overflows). Also, scripts written in C compiled language are preferred to scripts written in interpreted language, because interpreted script source is readable. Moreover, it is forbidden to let any shell interpreter in the WWW server chrooted environment because it gives the possibility to execute shell command in case of security hole.

Secure C coding standards have been developed at CNES. Moreover, secured on-the-shelf functions and programs have been developed by CNES and distributed to the WWW developers. The programs log in a dedicated log file all the abnormal requests, and every abnormal mode. For each user authorised to download files from the Storagetek, there is a quota on the WWW server machine in order to avoid denial of service.

Sensitive functions like CD-ROM order and WWW server registration are not manual but accomplished by the operators.

And last, the www server only has read rights on remote servers like Oracle and Storagetek.

EURIDIS

EURIDIS project

The Global Positioning System (GPS) is a worldwide radio-navigation system. It allows at any time a worldwide determination of position anywhere on or above the Earth. GPS was developed by the DoD (*US Department of Defense*) primarily for military purposes, so the DoD is intentionally degrading the positioning accuracy by introducing some errors into the clock, the orbital data, ... This accuracy (about 100-meter horizontal with 95-percent availability), and the GPS integrity and availability are insufficient for the civil aviation requirements.

EURIDIS is a joint CNES/DGAC (*Direction Générale de l'Aviation Civile: French Civil Aviation*) program that leads to improvements in the accuracy, availability and integrity of the GPS signal. The EURIDIS goal is, in the geostationary INMARSAT III AORE satellite footprint, to provide through INMARSAT III transponder a supplementary look-like GPS signal to Civil Aviation.

The supplementary information sent through the INMARSAT III navigation payload come from the Mission Control Centre (MCC) computations of data. The MCC handles data from the ground stations and from data collected at external centres (solar activity, orbit and health of the GPS satellite, geostationary satellite manoeuvre bulletin...). The MCC sends also some information about status of EURIDIS service to Inmarsat Navigation Centre or to the Air Traffic Control located at STNA (*Service Technique de la Navigation Aérienne: Air Navigation Technical Service*). MCC uses Internet and ISDN lines to obtain and deliver data.

EURIDIS Architecture

In such a project, the integrity of transmitted and received data is a strong requirement.

Introduction

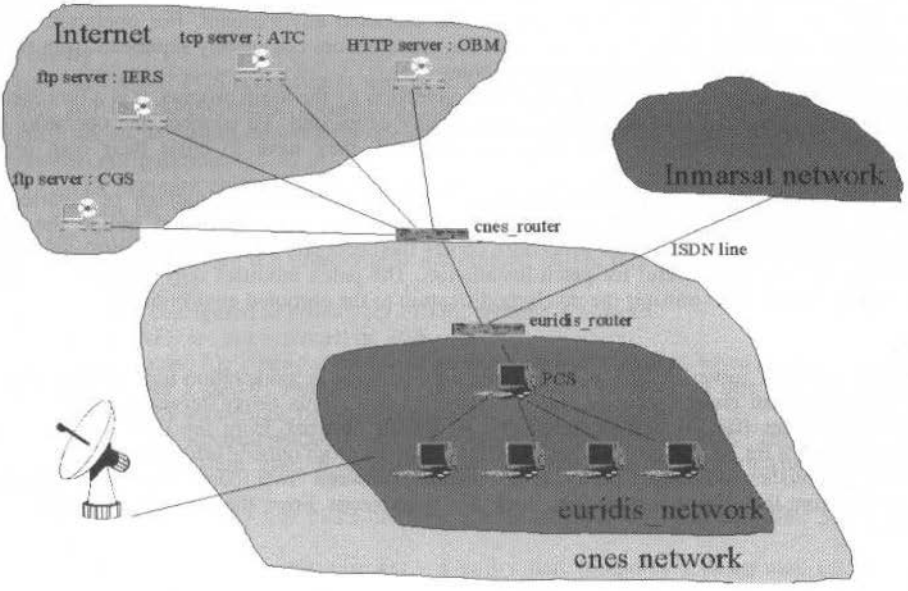


Fig. 4 EURIDIS architecture

To achieve this goal, Euridis has installed its own sub-network (euridis_network) designed upon the CNES private network (fig. 4). A specific router (euridis_router) filters the traffic flow between the two networks. In the same way, the router cnes_router filters the traffic flow between the CNES network and Internet.

Only necessary traffic flow can pass those routers. The computer called PCS (fig. 4) is the only one allowed to communicate with euridis_network outside. Moreover, PCS initiates all the communication to the outside; this means that only client software can be used to communicate with the outside.

In order to control the traffic flow with the outside, a SOCKS5 server has been installed (see session *Socks5*).

On the other hand, Euridis team expressed the need for the operators to work on PCS. The operators have to send and receive mails, and perform ftp and WWW transfers. The presence of operators on a computer which hosts a proxy and which can communicate with Internet needs very strong security requirements. In order to reduce the cost, the development is based on reused methods (usual CNES security methods) and in-house or COTS (*Commercial On The Shelf*) software (xlocktool, tiger, tripwire, tcp-wrapper, socks5, ...).

External Links

EURIDIS is connected to external servers, which are :

- International Earth Rotation Service (IERS) : FTP server that provides information useful for geostationary satellite orbitography (Polar motion, relationship between UT1, UTC, TAI)
- Air Traffic Control (ATC) : TCP server developed for EURIDIS. With this server ATC gets information from EURIDIS (operational state, GPS satellites status).
- Coast Guard Service (CGS) : FTP server that provides information about the GPS constellation (precise ephemeris, almanachs, GPS satellites status and prediction of satellite unavailability.
- Observatoire de Meudon (OBM) : HTTP server that provides information about solar activity and sunspot number.
- Inmarsat Navigation Centre (INC) : TCP server that provides to EURIDIS information about the geostationary satellite.

EURIDIS Security

Chroot

The operators are chrooted in a second environment, which has been rebuilt. This environment is restricted, a lot of programs have been suppressed (r-commands, administration commands, ...) and some have been "socksified" (ftp, netscape, ...). Only the administrators can connect to the core environment. On the other side, root is not able to connect in the chrooted environment. For operators nothing is changed, they can work as usual. When chrooted, the operators are completely separated from the rest of the disk, so they are separated from the socks5 server.

The chrooting is automatically operating at the connection by the login process. It is a functionality of the login process (system V), so nothing was specifically developed. To connect, the operators have to enter a common login-password. Then they are chrooted, they have to enter their own login and password.

The building of the chrooted environment was the biggest effort. The number of binaries and of devices has been limited as much as possible. An automatic procedure that builds the chrooted environment has been developed. This procedure copies the necessary files from the core environment to the chrooted one. It is very useful for patch installation. The patch modifies only the core environment. This procedure is a mean to transfer the patch modification to the chrooted environment.

Socks5

Socks5 is a proxy server. It acts as an intermediary between a workstation user and the Internet so that the enterprise can ensure security, administrative control. This proxy server is associated with a router (euridis_router (fig. 4) that separates the EURIDIS network from the outside network and a firewall server (PCS fig. 4) that protects the EURIDIS network from outside intrusion.

Socks5 establishes a secure proxy data channel between two computers in a client/server environment. From the client's perspective, Socks5 is transparent. From the server's perspective, Socks5 is a client.

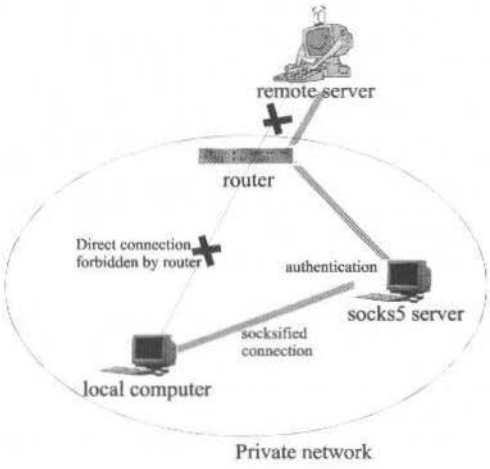


Fig. 5 socksification

Socks5 includes two primary components, the socks5 server and the socks5 client library. Socks5 client shared library dynamically socksifies existing network applications. When the client attempts to access the network, the client connects to the socks5 daemon instead of the application host (fig. 5). A shell script, called runsocks, socksifies applications, eliminating the need to recompile original applications. In fact runsocks makes the value of the environment variable LD_PRELOAD equal to the shared library path. So, the system loads socks5 shared library before executing any program.

Socks5 requires a username/password authentication. Each user must set the environment variables SOCKS5_USER and SOCKS5_PASSWD. A configuration file contains the information the server needs to determine if a user can connect to one port of a host. Then Socks5 server authenticates the requests, establishes a proxy connection, and relays data.

Before adopting Socks5 product, Socks5 server possibilities were tested. Moreover, the socks5 server security has been reinforced, we did some modifications :

- the socks5 passwords are now crypted like Unix passwords,
- the socks5 server is chrooted in its own small environment, so any attack against the socks5 server cannot corrupt the rest of the disk.

UNIX Security

The configuration of PCS is consistent with CNES strongest security standards: permissions mode of system files and directories, audit, accounting, file integrity with Tripwire, configuration check with Tiger, control access to various network services through the use of access control list with tcp-wrapper...

- System configuration
  - UNIX standard permission modes of files and of directories have been modified in order to increase security.
  - the environment variable TMOUT is set to a certain number of seconds. This means that any shell will terminate if a command is not entered within this number of seconds after issuing the PS1 prompt
  - System is configured to prompt for PROM security password.
  - NIS and NFS are not activated on PCS
  - The inetd.conf file contains the list of servers that inetd invokes when it receives an Internet request over a socket. Ftpd is the only server that is declared in this file. So, nobody can connect PCS with telnet, because the daemon telnetd is not declared.
  - Solaris 2.5.1 Recommended Patches are regularly installed on PCS. These patches fix the most critical system, user, or security related bugs that have been reported and fixed.
- COTS software
  - xlocktool is a utility that locks the screen against unauthorised access after 30 minutes idle period.
  - Tripwire is a file integrity checker - a utility that compares a designated set of files and directories against information stored in a previously generated database. Added or deleted files are flagged and reported, as are any files that have changed from their previously recorded state in the database. Using Tripwire, system administrators can conclude with an

extremely high degree of certainty that a given set of files and directories remain untouched from unauthorised modifications, provided the program and database are appropriately protected (e.g., stored on a DAT which is closed in a safe).

- Tiger is a package consisting of Bourne Shell scripts, C code and data files, which is used for checking for security problems on a UNIX system. It scans system configuration files, file systems, and user configuration files for possible security problems and reports them.
- Tcp-wrapper monitors incoming requests for telnet, finger, ftp, exec, rsh, rlogin, talk and other services. Operation is as follows: whenever a request for service arrives, the inetd daemon is tricked into running the tcpd program instead of the desired server. tcpd logs the request and does some additional checks.
- Socks5 is a proxy server which is described in part Socks.

- FTP server
  - Ftpd is the only service on PCS. This ftpd server is also chrooted in the operators' environment. Moreover we use tcp-wrapper: only one computer of euridis_network can connect.
  - Each FTP session is logged to the system log daemon syslogd which reads and forwards system messages to appropriate log files.
  - Only one login can connect by ftp. This login cannot connect to PCS.
- Shared mail
  - The operators have to work 24h/24h, so they have to share some information, for example they have to share the mail. For security reasons, each operator must have his own account on PCS, so we have developed a system of shared mail. This system also moves the mail from the core environment (where it arrives) to the operators environment (where the operators are) (fig. 6).
  - A user euridis has been created in both environments. In those two environments, none of these users can connect to PCS. Each time euridis receive a message, a shell script is executed and copies it into the /var/mail/euridis file of the chrooted environment.
  - In the chrooted environment, a program called ERDmailtool allows the operators to access the shared mail. This program is owned by the user euridis and the sticky bit is set (When the SUID (set user ID) bit is set on the permissions of a binary file, it executes with the UID of the owner rather than that of the person executing it. An SUID binary has access to all the files, processes, and resources belonging to the owner of the binary file.). This sticky bit is necessary because one must be owner of a mail file to read and modify it. The mail file in the chrooted environment belongs to euridis user and to the operators group, so the operators can read and write in this file.

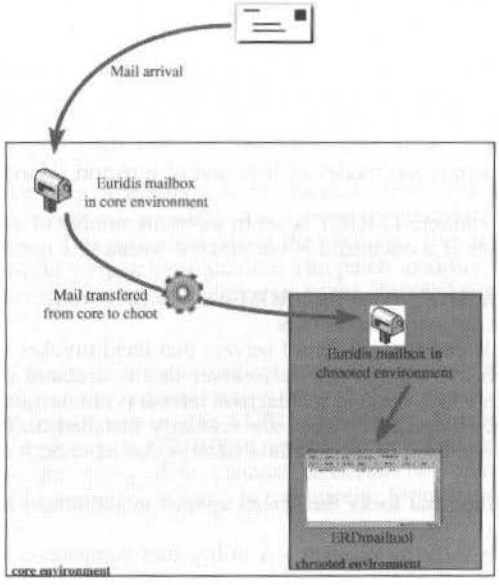


Fig. 4 shared mail on PCS

The operators can also send mail to the outside. They can use ERDmailtool and also just mail or mailtool.

Conclusion

These two projects have got common characteristics which could be applied to nearly all ground segments connected to Internet.

First, physical security should not be forgotten.

Logical security is also essential :

- traffic flow filtering by the way of routers
- chrooting is very interesting because it protects the computer core from any software security hole.
- coding standards and source quality is also essential, because they permit to reduce the number of bugs, so the number of security holes.

But it shall be pointed out that security has a price and induces lot of delay. So a compromise has to be done.

Acknowledgements

I would like to gratefully acknowledge the people who have contributed to EURIDIS and ScaRaB projects. Henry MARQUIER, H. SECRETAN and N. SUARD for their help and advice, O. MAS and P. GELARD for technical aspects and M. MIQUEU, G. SOULET, O. PERIE and S. PECHMALBEC for security aspects.

References

[1] WWW server which presents the ScaRaB project and which provides data to scientific community. http://scarab.cnes.fr:8020

[2] Design & Operations of the EURIDIS System, P. Gouni, D. Flament (Thomson-CSF), H. Secretan, A. Job (CNES), DSNS'96, St Pétersbourg

[3] EURIDIS Software, Critical Functions, Main architecture and Quality Assurance. N. Suard (CNES). DASIA 97, Sevilla, Spain 26-29 May 1997

[4] EURIDIS External Links and Security Aspects. R. Moreno (CNES) DASIA '98 conference on 'Data Systems in Aerospace', Athens, Greece, 25-28 May 1998.

[5] Doctrine générale du CNES pour la sécurité des systèmes d'information. 26.91/DG/IG/SI. Ed 1, Rév. 0 du 10/07/1992.

[6] Exigences de sécurité pour l'ouverture des réseaux IP du CNES CT/AS/SE/SI 94-019 Ed. 2 Rév. 1 du 19/01/1994

[7] Introduction to socks http://www.socks.nec.com/introduction.html

[8] UNIX Solaris 2.5.1 manual pages.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 237

ISSN 0100-7386
Printed in Brazil

Past & Present Experiences

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp.238-242

ISSN 0100-7386
Printed in Brazil

Evolution of Hubble Space Telescope Operations: Best Practices and Lessons Learned Over Eight Years

Julio L. Marius
NASA
Goddard Space Flight Center
Greenbelt, Maryland, USA 20771
jmarius@hst.nasa.gov

Jack E. Leibee
NASA
Goddard Space Flight Center
Greenbelt, Maryland, USA 20771
jleibee@hst.nasa.gov

Abstract

The Hubble Space Telescope has been successfully operating and providing world class science for the past eight plus years. Significant increases in science productivity and efficiency have been made during this time. Many of these increases and other operational improvements are attributable to changes in operations management and processes. This paper describes four especially significant changes in ground operations: Hubble Space Telescope Anomaly Reports; Flight Readiness Reviews; an HST web page; and changes in Flight Operations Teams at the Space Telescope Operations Control Center. These changes have improved overall operations efficiency, reduced operational risk, and reduced costs. The processes can be applied to current and future missions.
Keywords: *Hubble Space Telescope Operations, Operations Efficiency, Ground Support.*

Background

Operations of the Hubble Space Telescope (HST) have evolved over the years to accommodate new scientific instruments, loss and/or degradation of spacecraft hardware, and unexpected performance of some spacecraft/ground systems. Due to spherical aberration and other problems, the early years were marked by manpower-intensive operations in a reactive mode. The HST Project and its contractors have overcome most of these problems via innovative management practices, process changes, and ground/spacecraft software improvements. Spacecraft observing efficiency has more than doubled, and staffing has been reduced. Because HST is refurbished in orbit using the Space Shuttle, an extensive engineering trending and analysis capability is required to ascertain whether spacecraft subsystems can last the lifetime of the mission (20 years) or should be replaced during the servicing missions. This capability is also helps manage spacecraft assets carefully to ensure their longevity.

Organizational, process, and system changes have had to be made frequently to maintain the high productivity of HST. Continuous Process Improvement techniques have been successfully applied to reduce operational costs, including combining disparate systems into a single cohesive system. Processes have been developed to ensure that the multitudes of flight software and ground system changes that have to be made are implemented correctly.

Since the launch of the Hubble Space Telescope, four major initiatives have significantly reduced the risks inherent in operating a multi-billion dollar spacecraft:

- the development and use of an anomaly tracking system called Hubble Space Telescope Anomaly Report (HSTAR);
- a process called Flight Readiness Review (FRR);
- a web page for the HST project; and
- changes regarding Flight Operations Teams (FOTs) in the Space Telescope Operations Control Center (STOCC).

HSTARs

The tracking of problems and their resolution has required a significant effort. Over 7000 problem reports have been written since launch. They have been written on problems with spacecraft, ground system, procedures, configurations, documentation, and even personnel behavior.

The HSTAR system supports the rigorous process of identifying, resolving, and closing out problems. Initially, HSTARs were written against a problem and submitted daily with any relevant

documentation. The documentation, however, was not in electronic form. A system engineer not only assigned the HSTAR to an organization or individual for analysis, but also assigned a criticality level: critical, urgent, or routine.

A critical HSTAR requires almost immediate response since the health and safety of the spacecraft or the ability to continue science operations could be at risk. Whenever the spacecraft entered safemode or the timeline was interrupted, a critical HSTAR was always written. In many instances, workarounds were implemented while either flight or ground system software was modified to correct the problem.

Sometimes these flight/ground system changes had to be made as a result of spacecraft hardware failing or degrading.

Urgent HSTARs are written when operations could continue safely, but there was an increased risk of failure. From a software fix standpoint, new deliveries were usually made within weeks.

Routine HSTARs are written against any operational failure, such as the inability of the spacecraft Fine Guidance Sensor to acquire a binary star as a guide. In this case, the guide star would be removed from the catalog, and the HSTAR closed.

Closure of HSTARs is done by a board consisting of members of the various elements of the ground system, operations, and the spacecraft. The board meets biweekly and reviews the recommended closure. A cognizant board member reviews the closure prior to the meeting and speaks to any questions from other board members. In some instances, the person who analyzed the HSTAR may attend the meeting if the closure requires further technical depth. Some recommended closures have been rejected due to insufficient information or even an incorrect fix. This latter example is why having a board membership which crosses organizational and technical boundaries is so important.

HSTARs were originally tracked electronically in terms of HSTAR number, originator, title, and date. The detailed information and supplemental data were stored in binders. No cross-reference to other HSTARs was possible. This problem has been corrected with the new HSTAR electronic system. The HSTAR system now allows cross-referencing by originator, type, key words, and system.

Weekly reports are produced which apprise management of the frequency of occurrence of HSTARs and the suspect systems. This is used as a gauge for to determine how operations are proceeding and, more importantly, to identify degradation of any spacecraft components. Along with the extensive trending program, the weekly HSTAR reports help management determine whether any hardware needs replacement during the next servicing mission. In servicing missions, which are scheduled to occur approximately every 3 years, the Space Shuttle captures HST, replaces degrading or failed components, and upgrades instruments and other hardware.

FRR

The Flight Readiness Review (FRR) evolved after some early minor mishaps that occurred following the HST launch. Due to the multitude of problems encountered post-launch, including spherical aberration and solar array disturbances impacting vehicle jitter, several engineering tests were conducted and flight software changes were made. Engineering level reviews of the test procedures and/or flight software changes were required, but the contents and thoroughness of the reviews varied. The HST Project quickly realized that a more formalized mechanism was needed to ensure complete success of the activities. This resulted in the formalized FRR.

For each activity, a test conductor is assigned to lead all the planning, development, and execution activities. This person is usually a member of the Systems Engineering Organization and a contractor. The team is composed of those people who design the test, implement the changes, build procedures, test the processes, analyze the data, conduct the test, and any other associated support.

A pre-defined FRR package outline was established which identifies the areas which must be addressed when the FRR is conducted prior to the test. Major items in the presentation package include:

- background of problem;
- Test Team identification and roles & responsibilities;
- problem solution;
- constraint review;
- script timeline execution;
- contingency planning;
- any issues/concerns; and
- script review.

Action items are tracked and closed before test execution. Although this process may appear to be standard, it is such a rigorous process that it requires internal dry runs with the Test Team prior to being presented to the HST Project and its contractors.

Because more than 250 flight software installations have occurred since launch, the FRR process has had to undergo some refinement. Early in the mission, the flight software changes were being made by a

contractor at a facility in California. It was difficult to communicate effectively all the necessary information needed for successful flight software changes. Although only a couple of errors occurred, none with any serious consequences, it was decided to add more structure to the process. To protect against any unexpected consequences of a new flight software change, a back-out (contingency) load was always generated which could be uplinked to restore the flight software to its pre-installation configuration. The formal FRR process was also used, resulting in a higher confidence level for success.

As a result of these changes, the amount of time required to uplink flight software and verify functional performance was reduced significantly. Early in the mission, up to 4 hours of on-board regression testing was used to confirm the effectiveness of a change. After changing the process, no on-board testing was needed because the level of confidence was now near 100%. Since HST observing time is limited and scientific observations are precluded during installation time, a significant increase in scientific observation time was another benefit of the new FRR processes.

HST Web Site

A third area of improvement in operations and management was the development of a web page for the HST Project. This began in 1993 and has expanded greatly since. The web site has many uses, but its most important ones are:

- worldwide access to all HST reports (limited only by security);
- spacecraft subsystem engineering trending status;
- servicing mission planning; and
- servicing mission real-time procedure approval.

This capability has allowed engineers and managers from the USA and overseas to find out the status of HST operations at a moment's notice. Access is limited and security enforced via the standard user ID, password, and IP address. Special engineering reports are published on the web site and accessed by engineers from locations remote from Goddard Space Flight Center (GSFC). During anomaly investigations, the web site has allowed experts to analyze data from their home sites.

HST servicing missions last from 7 to 10 days. Unexpected conditions during the mission may cause plans to change rapidly, and the web site has been used to post the most recent and up-to-date plans. Although there is much discussion via telephone, the web site ensures that accurate information and data are available to everyone on the ground, regardless of physical location. The web page facilitates communication during servicing missions, for example, between the HST Project's senior management at Johnson Space Center (JSC) and the HST command and control personnel at GSFC.

The HST servicing mission timeline and command plan, plus all of the contingency plans, are available on the web with the most recent updates. This way, engineers and managers at GSFC and JSC do not need to rely on e-mail or faxes to find the latest information. All managers and engineers have immediate access to this web page at their consoles or desks. In the past, electronic transfer of updated products was tried, but sometimes resulted in confusion as to which version to use. Under the new system, everyone is notified via e-mail when a new product is delivered. Frequent reloading of the URL ensures all the users are in lock step with one another.

Because of the high number of changes which occur during a servicing mission, Operations Requests are required to document any deviation from the planned script. Prior to the implementation of the web site, approval from a remote site such as JSC required that the Operations Request be faxed and distributed to the appropriate people. With the new system, notification of Operations Requests is electronically sent via e-mail to those persons whose approval is required. They then view the requests on the web and can electronically approve/disapprove it; approval summary information is provided on the web site. This has significantly sped up the approval process. In addition, the status of all Operations Requests is posted, thus helping the Mission Operations Manager determine the readiness of any individual request. Timeliness is extremely important due to the astronauts' limited EVA time, and the difficulty of some of the decisions which have to be made.

FOTs

The Space Telescope Operations Control Center STOCC facility is located in the GSFC compound and consists of several mission operational areas.

- Mission Operations Room and Servicing Mission Operations Room (MOR/SMOR): In this facility resides the Flight Operations Team (FOT) that is responsible for the health and safety of the telescope.
- Data Operation Control (DOC): This is the front-end processing facility for telemetry and engineering data. It is also the center for all data communications and ground system configuration.

- System Engineering and Evaluation Room (SEER): This facility is used for system engineering testing and evaluation of spacecraft anomalies and resolutions.

The MOR/SMOR and the DOC operate twenty-four hours a day, seven days a week. Around the clock staffing is provided by four teams in three shifts. The STOCC was initially configured and staffed to meet requirements for the deployment of the telescope and the support of the scientific mission. However, after the initial on-orbit validation and verification, it was necessary to streamline the operations and servicing support for the telescope.

STOCC Evolution

The STOCC was configured along the lines of traditional NASA/GSFC missions, one console engineer per subsystem. The STOCC personnel consisted of professional engineers with high levels of training and experience. Each deployment FOT had seven console engineers monitoring the different subsystems onboard the telescope. Once operations became routine, the FOT engineering functions were consolidated, and teams were downsized from seven console engineers to five. A second downsizing of the FOTs, from five engineers per team to three, accommodated mission operations requirements to prepare and plan the first servicing mission.

Functional FOT Positions

Historically, there were seven functional positions in FOTs.

- Shift Supervisor (SS): Responsible for the overall activities of the team on duty. This position was the main point of contact for any issues related to the ground system and the telescope.
- Operations Controller (OC): Responsible for all command activities and the management of the tape recorders onboard the telescope.
- Pointing Control Subsystem (PCS): The complexity of the onboard navigational system required a dedicated position.
- Data Management Subsystem (DMS) and Instrument Communications: All communications, data formats and computers on board the telescope are handled by this position.
- Electrical Power Subsystem (EPS): The solar array, batteries and power distribution interfaces are handled by the position.
- Science Instrument (SI) Responsible for all the scientific payloads on board the spacecraft, as well as the execution of all science activities.
- Optical Transport Assembly (OTA): The initial problems with the secondary mirror required minor focus adjustments, and management of fixed star trackers and fine guidance sensors.

The initial optical problem of the mirror and the complexity of HST required careful monitoring of all subsystems and close evaluation of subsystem performance. Thus, in addition to the FOT on duty, there were experienced subsystem engineers supporting the FOT by testing the different subsystems onboard.

As the mission matured and the there was a better understanding of all spacecraft behaviors, mission requirements were reevaluated to optimize the combination of operational functions. It was also necessary to determine the minimum team size that would not place any mission objectives at risk. There were several issues that became obvious during the evaluation process. Several of the most critical issues follow.

Skill Mix and Level of Experience

Most of the FOT engineers had different engineering backgrounds, as required by the telescope subsystem, but

Table 1 FOT Position Changes

Deployment (Seven Positions)	Operations (Five positions)	Consolidated (Three Positions)
SS	SS	SS/CC
OC	OC	
PCS	PCS/SI	PCS/OTA/SI
DMS/I&C	DMS/I&C/EPS	DMS/IC/EPS
SI	SI/OTA	
EPS		
OTA		

with common spacecraft operations experience. Engineering communication problems were solved by providing additional training, hands-on experience, and reassigned position.

Work Load Distribution

An equitable load distribution was necessary to avoid any perception of unnecessary burden when additional work was required.

Training and Certification

The training program was revamped to accommodate a multiple discipline approach. The off-line subsystem engineers were given the responsibility for the additional training and certification of the FOTs and a mentoring program was established. Table 1 shows the final consolidation of all flight operations personnel. The consolidation did not cause any individual to lose his/her job. They were reassigned to support servicing mission activities and other preparation activities. This transition offered the advantage of having well prepared and experienced mission operations engineers in the preparation and actual performance of the servicing missions. The consolidation of FOT positions were successful and was used as a model in consolidating other HST support organizations and facilities.

Conclusion

The history of HST operations provides insight to spacecraft management and operations practices and processes. Because much of the scientific success of HST can be attributed to these practices and processes, they should be considered for other missions, large or small.

HSTARs are written whenever a problem occurs, regardless of the type of problem. FRRs are held whenever an engineering test, flight software installation, or other non-nominal activity is to occur. The web page allows worldwide dissemination of current HST status, and facilitates immediate response from remote personnel. STOCC changes consolidated and streamlined operations.

The HST Project continues to pursue better operational practices and processes to reduce costs, reduce risks, and maximize the scientific return of HST.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp.243-247
ISSN 0100-7386
Printed in Brazil

Leop Operations of SCD2, INPE's Second Environmental Data Collecting Satellite

Pawel Rozenfeld*
Valcir Orlando
Roberto Luiz Galski
INPE - Instituto Nacional de Pesquisas Espaciais
Av. dos Astronautas, 1758 - CP 515
12201-970 - São José dos Campos, SP Brazil
pawel@beta.ccs.inpe.br

Abstract

The SCD2, the second environmental data collecting satellite developed by the Brazilian National Institute for Space Research (INPE), was launched on 22 Oct 1998, by the American Pegasus launcher. The paper presents, at first, a brief overview of the INPE's ground system facilities. The main ground system improvements, and the main differences between SCD2 and SCD1, as well as its implications on the LEOP planned activities are presented and discussed. The nominal timeline of the developed Flight Operation Plan (FOP) for LEOP is commented, including the process of rehearsals, simulations and training the operational team in the FOP execution. Finally, the post-facto occurred LEOP of the SCD2 is described, outlining the main problems which has been faced, the adopted solutions and the main divergences from the nominal planned case. In conclusion, the lessons learnt with this second experience of INPE operational structure in supporting a satellite LEOP phase is highlighted.
***Keywords**: SCD2, LEOP Operations, Environmental Data Collecting.*

Introduction

After almost 6 years of successful in-orbit operation of the satellite SCD1 designed for one year in orbit life, a new satellite has been launched by INPE on 22 Oct 1998. This satellite is SCD2. Its aim is to improve the performance of the existing Environmental Data Collection Mission, a big success among the users of this system, as well as to extend the useful life of the mission. SCD1 in-orbit life has exceeded its nominal life, making any forecast of its future performance an unknown. The SCD2 LEOP operations, (Orlando, V., 1998) are described in this paper.

Section 2 recalls INPE's Ground Control System, (Rozenfeld, P., 1994). Section 3 describes the Data Collecting System (Yamaguti, W., 1994), which make use of SCD1 and SCD2 and presents the trend for expansion of this system caused, mainly, by excellent performance of SCD1 in orbit. An overview of Data Collecting Satellite series composed by SCD1 e SCD2 is presented in section 4. Main differences between the two satellites are emphasized.

Next, the Ground Control System preparation for SCD2 operation are presented in section 5. SCD2 LEOP operations are described in section 6. In-orbit tests and initial SCD2 performance evaluation, showing great improvement in data collecting system performance is described in section 7. The conclusions are given in section 8.

Inpe's Ground Control System

INPE's Ground Control System, which controls at the present time SCD1 and SCD2 satellites, is composed of a Satellite Control Center (CCS) located in São José dos Campos and two Ground Stations. One is located in Cuiabá at the geodetic center of South America, whose visibility circle covers it almost wholly. Another one is located in Alcântara, near equator, at the Brazilian Launching Center. A dedicated communication network connects the three sites. The 2 Ground Stations, besides being the TT&C stations, are, also, Data Collecting Platforms (DCPs) receiving stations.

Data Collecting System

The mission of SCD1 and SCD2 satellites is to collect environmental data by means of automatic Data Collecting Platforms scattered through out the Brazilian territory. DCPs are unattended terminals which collect the data and uplink them in 400MHz (UHF band) to the satellite. When the DCPs, the satellite and a receiving station are in mutual visibility the environmental data are downloaded in S-band to the receiving stations (Cuiabá and Alcântara). These data are recorded at the receiving station and are transferred soon after the pass to the Data Processing Facility located in Cachoeira Paulista, near São José

dos Campos. They are processed and made available to the users by means of Internet every 0,5 hour. The efficiency of this system makes it very popular among the users.

At the time of the SCD1 launch 6 years ago, there were about 20 DCP available for this mission. However, due to excellent performance of SCD1, at the time of the SCD2 launch there were 270 operational DCP, some 40 are in maintenance, 110 being accepted and 232 being purchased. There was tenfold increase of the DCPs. Altogether, it is expected to have about 700 DCP by the end of this year, meaning a 35fold increase of DCPs in use.

The DCPs have several applications such as meteorology, hydrology, geomagnetism, atmospheric chemistry, tide monitoring, tropical forest regeneration monitoring and so on.

Data Collecting Satellites Series

The Data Collecting Satellites (SCD1 launched on Feb 9, 1993 and SCD2, launched on Nov 22, 1998) are small satellites (about 100kg), in low earth orbit (750km altitude), almost equatorial (25° inclination to cover the Brazilian territory), active attitude control and spin stabilized, with no orbit control. Their on-board subsystems are Structure, Power Supply, On-board Data Handling (Computer), Housekeeping Telecommunications, Data Collecting Payload, Attitude Control, Thermal Control and some Experiments.

Although, basically, SCD1 and SCD2 are similar, the experience gained from SCD1 in-orbit operation caused different implementation of some on-board subsystems, aiming at better mission performance.

The first problem with SCD1 was the existence of a silence zone, caused by the interference of the radiation patterns of the top and bottom antennas of the same polarization (RHC) at nearly 90° to the satellite spin axis. This silence zone makes it difficult to receive the telemetry and DCP data as well as it shall be avoided when performing ranging sessions and sending telecommands.

This effect was avoided in SCD2 by using different antenna polarizations (LHC and RHC). Therefore, SCD2 S-band antennas for TT&C (transmission and reception) and DCP (transmission) at the top face were LHC polarized while at the bottom face were RHC polarized. At the top face there were located also a high performance UHF DCP receiving antenna with RHC polarization.

Because of the existence of these antennas on the top face, SCD2, differently from SCD1, has no solar cells there. This generated a new, compared to SCD1, constraint on SCD2 attitude. The sunlight could not inside on the top face because it could cause a thermal problem.

Also, absence of the solar cells on the SCD2 top face caused lower on-board electric power generation unless the satellite is keep at $90° \pm 10°$ to the ecliptic plan.

SCD1 is a spinner with no spin control. The launcher imprinted the initial spin rate of 120 rpm. The present days spin rate is 49 rpm.

SCD2 has an autonomous spin control that keeps the spin rate in the 32 to 36 rpm range.

The attitude control of both satellites (SCD1 constraint is 60° to 90° sun attitude while SCD2 constraint is 80° to 100° sun attitude) is done by means of a magnetic coil activated and de-activated by ground control. The more stringent SCD2 attitude control causes more frequent attitude maneuvers compared to SCD1. But the magnetic coil on board of SCD2 is twice that of SCD1, making shorter the duration of the attitude maneuver.

Ground Control System Preparation

In terms of Ground Control System availability for SCD2 control, the main impact was caused by two polarizations tracking of this satellite. It is recalled that SCD1 has the same polarization (RHC) of both on-board antennas. Therefore, the tracking by the Ground Stations was done in only one polarization. For SCD2 it is necessary to track in two polarization: LHC and RHC. Therefore, an upgrading of the antenna tracking systems at Cuiabá and Alcântara Ground Stations, by including a new tracking channel, was performed.

Also the reception of DCP data by the Ground Stations is done in two polarizations. Though the Ground Stations have 2 data channels for DCP reception, on order to avoid manual switching from one channel to the other, a diversity combiner has to be installed to automate the reception in two polarizations.

As far as the uplink channel is concerned, there is no way to avoid the manual switching of the polarizations. The instants of polarization switching were computed based on SCD2 nominal attitude and included in the pass predicts. Therefore telecommanding and ranging are done before or after this instant.

Having as the aim the improvement of the performance of the Data Collecting Mission done by SCD1, already in orbit, two constraints on SCD2 launch windows were imposed. The first one was to

make SCD1 and SCD2 passes over a Ground Station complementary. SCD1 passes over a Ground Station have a 10h gap with no visibility by the Ground Station causing the same gap in data collecting, as a consequence. So, the SCD2 has to fill in this gap in a way that a 24h data collection could be achieved. For this purpose, the difference between the right ascension of the ascending node of the satellites shall be about 180 degrees.

Another constraint is that when SCD1 and SCD2 passes are close to each other, they are not overlapping in such a way that operationally they are not interfering. For this purpose, the launch window has been chosen so as the difference between the latitude argument ($M+\omega$) of the satellites be less than 120 degrees.

As far as the operational personnel training is concerning, SCD2 software simulator developed at INPE was used. Due to a good experience in operating SCD1 this training was not very extensive and it was directed to the main differences between SCD1 and SCD2.

A Flight Operation Plan, to support the launching and early orbit phase (LEOP) of SCD2 was prepared. It was based on nominal injection point supplied by Orbital Science Corp and propagated thereafter. The SCD2 two first passes, similarly to SCD1, are visible by Alcântara Ground Station only. From the third pass on Cuiabá Ground Station comes into play.

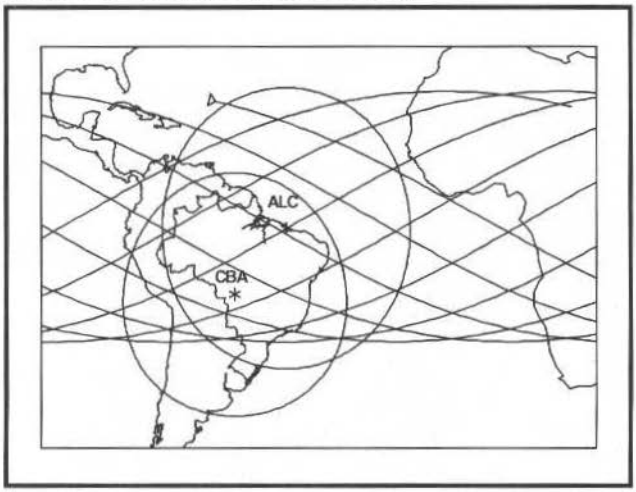


Fig. 1 First Orbit Ground Tracks

It was decided, also, not to control SCD1 during the first 24h after SCD2 injection in order not to overload the flight control team.

SCD2 LEOP

The Orbital Science Corp Pegasus three-stage launcher inserted SCD2 into orbit. This launcher is carried by a L1011 plane to a defined drop point over Atlantic where, after 5sec free-fall, the launcher first stage is ignited. After about 11min, the satellite is injected into its orbit.

The SCD2 launch activities started on Oct 22 five and half-hours before the satellite drop. Several confirmations of the Ground Control System readiness have been asked by the Launch Control Center. Some selected SCD2 telemetry data under the Pegasus shroud were available at the Satellite Control Center (CCS) showing good health of the satellite.

The L1011 took off from the Cape Canaveral Air Force Base on 22 October, at 23:05 UTC. During the flight the same SCD2 telemetry data were continuously available at CCS still attesting its good health. At 00:12:57 the separation of SCD2 from the Pegasus third stage occurred and the satellite was injected into its orbit.

Approximately 12 seconds after injection, SCD2 entered into the visibility circle of Alcântara Ground Station. The signal acquisition occurred using the acquisition antenna indicating that the on-board transmitter was turned on by the separation switch. Good received signal level allowed immediately to switch over to the main antenna. The received telemetry was transferred in real time to the CCS in São José dos Campos. The analysis of the telemetry showed that the satellite had no problem on-board. Therefore, following the procedures of the Flight Operation Plan, several ranging sessions have been performed. During this pass there was no antenna polarization switch over.

Five minutes after separation, OSC sent to CCS the SCD2 injection state information. Using this information as the initial conditions and the ranging data, new pass predicts for the Ground Stations have been generated. These pass predicts were good enough to track SCD2 by Alcântara and Cuiabá Ground Stations with no problem. From that point on using the ranging data, the orbit determination has been refined. At the end of the eighth pass, the orbit has been satisfactorily determined and could support several days of sufficiently accurate pass predictions over the ground stations.

During the second pass over Alcântara the magnetometer have been turned on. At the same time, the sun aspect angle, as measured by sun sensors, have been checked. The thermal control required that the sun aspect angle should lie in the range $90°\pm10°$. The measured sun aspect angle was 90.3°, well in the middle of the allowed range. As far as the spin rate is concerned, the measured value was 37.8rpm, intentionally left above the nominal range (34 ± 2 rpm)

The other satellite parameters at the injection were:

Apogee: 773.4km (real)
750.0km (nominal)
Perigee: 747.2km (real)
750.0km (nominal)
Inclination: 24.9° (real)
25.0° (nominal)

After working for 19h, the flight control team went to rest.

On the second day the SCD2 DCP Transponder have been turned on because the power supply system showed nominal performance and the On-Board Computer memory load started. This was done in two consecutive passes, much quicker that in the case of SCD1. Also, SCD1 control has been resumed.

After this successful loading and fine attitude determination SCD2 was declared to be able to go to the in-orbit acceptances phase, which showed the nominal performance of the redundant equipment. Thereafter the routine phase have been declared.

The Figure 2 shows the position of the two satellites at the SCD2 injection time.

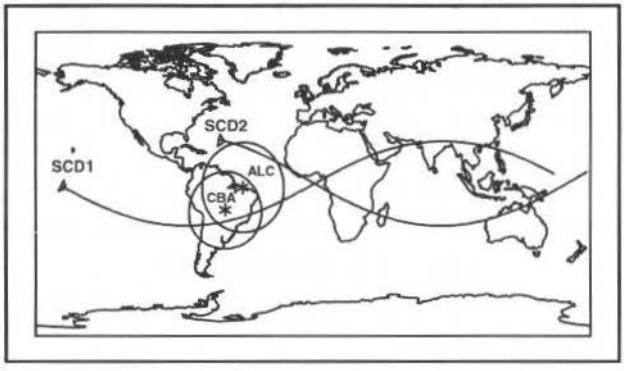


Fig. 2 Phasing Between the Satellites

SCD2 Performance Evaluation

Two months after LEOP, the performance of all SCD2 subsystem are within the nominal ranges.

The Power Supply subsystem shows a nominal performance. The in-orbit test of the redundant equipment presented nominal results. All the charge and discharge units performed nominally.

The On-Board Computer has been loaded on the second day after launch and has been operational up to 17Nov1998. At that day it has been turned off in order to avoid meteorite shower and turned on the next day with no problem whatsoever. The test of time tagged telecommand have been performed with success.

The Housekeeping Communication Subsystem has nominal performance. During the tests it was detected a problem with the redundant transponder which is being investigated by specialists.

The Attitude Control Subsystem was tested during in orbit tests with satisfactory performance. The autonomous spin rate system was telecommanded to decrease and increase the spin rate. When it reached 36rpm the On-Board Computer switched the system off.

The first spin axis re-orientation maneuver was performed with success on 28 Dec 1998. When the sun aspect angle were close to 99°. The magnetic coil was on until the sun aspect angle reached 91° when it was turn off, as it can be seen in Fig. 3.

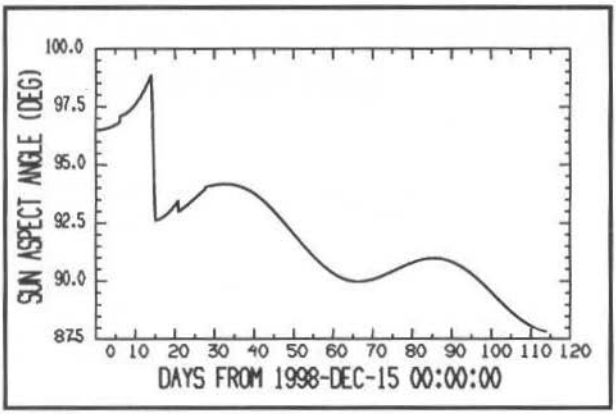


Fig. 3 Sun Aspect Angle

From that point on the simulation showed that the sun aspect angle would reach 85° by middle February and then would starts to grow. The next attitude maneuver should occur next April.

The Data Collecting Subsystem presented results extremely satisfactory. With the beginning of SCD2 operation as a Data Collecting satellite, the Data Collecting Mission performance using SCD1 and SCD2 increased more than twice as compared to SCD1 only performance.

SCD2 has a performance 30% better that SCD1 as far as the number of DCP messages in concerned. This performance has been expected because of the improvement in SCD2 design as compared to SCD1. This trend shall be kept by means of a larger number of SCD2 attitude control maneuvers.

Conclusions

The launch of SCD2 on 22 Nov 1998 represents a success in terms of making its Data Collecting Mission an operational one.

Its represents, also, the maturity of the INPE's Ground Control System and its readiness to assume more complex mission foreseen for near future.

References

Orlando, V., Kuga, H. K. and Guedes, U. T. V., 1998, "Flight Dynamics LEOP and Routine Operations for SCD2, the INPE's Second Environmental Data Collecting Satellite", Advances in the Astronautical Sciences, Volume 100, Part II, pp. 1003-1013.

Rozenfeld, P., Orlando, V. and Schneider, E. T., 1994, "Overview of the INPE's Satellite Tracking and Control Center and Main Aspects of its Debut in Satellite Operations", Revista Brasileira de Ciências Mecânicas, Volume XVI, Special Issue, pp. 421-425.

Yamaguti, W., Ribeiro, E. A., Becceneri, J. C. and Itami, S. N., 1994, "Collection and Treatment of the Environmental Data with the Brazilian Satellite SCD1", Revista Brasileira de Ciências Mecânicas, Volume. XVI, Special Issue, pp. 205-211.

RBCM - J. of the Braz. Soc. Mechanical Sciences ISSN 0100-7386
Vol. XXI – Special Issue - 1999 - pp.248-250 Printed in Brazil

Dealing With Uncertainty When Managing an Earth Observation Satellite

Eric Bensana
Gérard Verfaillie*
*Office National d'Etudes et de Recherches Aérospatiales (ONERA - CERT)
2, avenue Edouard Belin, BP 4025, 31055 Toulouse Cedex 4, France
(eric.bensana ,gerard.verfaillie)@cert.fr

Claire Michelon-Edery
Nicolas Bataille
Centre National d'Etudes Spatiales (CNES)
18, avenue Edouard Belin, 31401 Toulouse Cedex 4, France
(claire.michelon-edery,nicolas.bataille)@cnes.fr

Abstract

The possible presence of clouds is the main origin of uncertainty when managing earth optical observation satellites. Forgetting it can lead to poor results in terms of really achieved photographs. In this paper, we show how a mathematical approach, drawn from the Markov Decision Process *framework, allows us to define a rational way of taking in account this uncertainty in the daily optimization process.*
Keywords *: planning, uncertainty, markov decision process*

Context

At the highest level, managing an earth observation satellite, like *Spot*, consists in choosing the sequence of photographs to be taken. Typically, this choice is made each day for the next day. The set *SA* of the photographs that can be taken the next day (according to the satellite trajectory and the instrument maneuvering ability) is extracted from the current order book. From this set *SA*, one tries to extract a subset *SE* that is feasible (there is no conflict between photographs in *SE*; all the physical satellite constraints are satisfied) and optimal (a gain, usually equal to the sum of the gains associated with each selected photograph, is maximum).

Due to the nature of the problem (a multi-knapsack problem with a large number of capacity constraints, each of them involving only a small number of 0/1 variables[1] and to the size of the instances to solve (until several hundreds of 0/1 variables), efficient algorithms, (Agnèse, J.C. et al.:, 1995, Bataille, N. et al.:, 1996) are needed. Whereas optimal algorithms, (Verfaillie, G., 1996), using a *Branch and Bound* schema, can solve small and medium size instances , only sub-optimal algorithms, using a *Iterative Local Search* schema, can deal with large size instances.

Unfortunately, this daily optimization approach does not take into account the fact that most of the photographs have several other feasibility opportunities after the next day and that the number of remaining feasibility opportunities is highly variable, depending on the deadline associated with each photograph.

Moreover, it takes into account, neither the uncertainty about the realization the next day of the selected photographs (due, with optical instruments, to the possible presence of clouds), nor the uncertainty about the number and the nature of the photographs that will be concurrently added to the order book.

To face this problem, gains associated with each photograph are usually modified in order to favor photographs, that have the smallest number of remaining feasibility opportunities and the highest realization likelihood (good meteorological forecast for the next day). But the way of combining the three criteria (gain, number of remaining feasibility opportunities, meteorological forecast) is not obvious and is generally empirically achieved, without any clear idea of the consequences in terms of really achieved photographs.

A rational approach

However, a mathematical approach, drawn from the *Markov Decision Process* (*MDP*) framework, (Puterman, M., 1994), currently used in *Decision Theory*, can help us to define a rational way of aggregating those three criteria.

---

[1] A 0/1 variable is usually associated with each photograph. The value 1 (resp. 0) means that this photograph is selected (resp. not selected).

To take into account the presence of several feasibility opportunities for a photograph, it is necessary to consider a global gain criterion over a given horizon rather than a daily gain criterion. A sensible choice consists in considering an horizon that covers all the feasibility opportunities of all the photographs belonging to the current order book. To take into account the presence of uncertainty, it is necessary to consider an expected global gain criterion rather than a global gain criterion.

Using this expected global gain criterion, the strict *MDP* approach leads us to intractable problems, because of the lack of knowledge, even in terms of probability, about the photographs that will be added to the order book and, above all, because of the huge number of states that should be explored by the *Dynamic Programming* algorithm, currently used in the *MDP* framework to compute optimal policies.

Fortunately, thanks to some simplifying assumptions (essentially, no influence of the current decision upon the future expected gains associated with the photographs that either belong to the current order book, or will be added to it), one can establish that the optimal policy (the one that maximizes the expected global gain) consists in selecting each day a set *SE* of photographs that is feasible and maximizes the sum of the weights of the photographs in *SE*, with weights set according the following formula:

$$w(p,\ d_c,\ \pi^*) = g(p) \times p_r(p, d_c) \times P_{ef}(p,\ d_c,\ \pi^*) \quad (1)$$

$P_{ef}(p,\ d_c,\ \pi^*)$ being computed by the rule :

if $RFO(p, d_c) = \varnothing$ (2)

then $P_{ef}(p, d_c, \pi^*) = 1$

else $P_{ef}(p, d_c, \pi^*) =$

$$\prod_{d \in RFO(p, dc)} [1 - p_r(p, d) \times p_s(p, d, \pi^*)]$$

where:

- p is a photograph;
- d_c is the current day;
- π^* is the optimal policy;
- $w(p,d,\pi)$ is the weight to be associated to the photograph p, the day d, according to the policy π;
- $g(p)$ is the gain associated with the actual realization of the photograph p;
- $p_r(p,d)$ is the realization probability of the photograph p the day d;
- $P_{ef}(p,\ d,\ \pi)$ is the non-realization probability for the photograph p on the days after d, using the policy π;
- $RFO(p,d)$ is the set of feasibility opportunities of the photograph p, remaining after the day d;
- $p_s(p,d,\pi)$ is the selection probability of the photograph p the day d, using the policy π.

The realization probability of a photograph p the day d can be easily obtained, either from short term meteorological forecasts, or from long term climate statistics.

As for the selection probability of a photograph p the day d, using the policy π, if one assumes that the order book keeps globally stable, at least over a large period, one can consider that it is a function f of p's weight, localization and type, *i.e.*

$$p_s(p, d, \pi) = f[w(p, d, \pi), l(p), t(p)]$$

where $l(p)$ is p's localization and $t(p)$ is p's type. Indeed:

- the higher p's weight is, the higher p's selection probability is;
- the higher the demand in p's area is, the higher the likelihood of conflict with other photographs is and the lower p's selection probability is;
- the more resource consuming p is (example: stereo demands), the lower p's selection probability is.

But, how to fix function f? It seems that a sensible option consists in learning it, in fact in approximating it, either off-line from simulations, or on-line from the observation of the system behavior, (Berstekas, D. et al. 1996, Sutton, R. et al. 1998). For example, a multi-layer neural network could be used for that. Whatever the technical option is, note that on-line learning has the advantage to allow the system to adapt itself to mid and long-term changes in the size or the nature of the order book.

As soon as the selection and realization probabilities (p_s and p_r) have been fixed, the recurrent equations 1 and 2 can be used to compute the weight to be associated with a possible photograph: the

process starts with the last opportunity ($RFO(p,d) = \varnothing$) and ends with the current one, alternating computations of weights and of selection probabilities.

Note that, as it was foreseeable, equations 1 and 2 favor photographs that:

- have a high associated gain;
- are subject to a good meteorological forecast for the next day;
- have a small number of remaining feasibility opportunities;
- are subject to bad meteorological forecasts for the days associated with these remaining feasibility opportunities;
- are localized in areas in great demand;
- are very resource consuming.

Conclusion

As a conclusion, thanks to some simplifications, an *MDP*-like approach provides us with a rational way of dealing with uncertainty when managing an earth observation satellite. The next step of this work would consist in fixing the learning process of the selection probability and in carrying out simulations in order to measure the actual gain in terms of achieved photographs.

References

Agnèse, J.C., Bataille, N., Bensana, E., Blumstein, D. and Verfaillie, G., 1995, "Exact and Approxinate Methods for the daily management of an Earth Observation Satellite". In Proc of the 5th ESA Workshop on Artificial Intelligence and Knowledge Based Systemes for Space, Noordwijk, The Netherlands. ftp://ftp.cert.fr/pub/verfaillie/estec95.ps

Bensana, E., Verfaillie, G., Agnèse, J. C., Bataille, N., Blumstein N., 1996, "Exact and Approximate Methods for the Daily Management of an Earth Observation Satellite". SpaceOps, Munich, Germany. ftp://ftp.cert.fr/pub/verfaillie/spaceops96.ps

Verfaillie, G., Lemaître, M., Schiex, T., 1996, "Russian Doll Search for Solving Constraint Optimization Problems". AAAI ftp://ftp.cert.fr/pub/verfaillie/rds-aaai96.ps

Puterman, M., 1994, "Markov Decision Processes, Discrete Stochastic Dynamic Programming". John Wiley and Sons.

Berstekas, D., Tsitsiklis, J., 1996, "Neuro Dynamic Programming". Athena Scientific, Belmont, Massachusetts.

Sutton, R. and Barto, A., 1998, "Reinforcement learning". MIT Press, Cambridge, Massachusetts.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 251

ISSN 0100-7386
Printed in Brazil

Operation Management

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp.252-259

ISSN 0100-7386
Printed in Brazil

Future Directions of Space Operations

Charles T. Force
VTEX International, Inc.
6411 Ivy Lane, Suite 650
Greenbelt, Maryland 20770, USA
ctforce@vtex.com

Abstract

This paper considers the future of space operations, focusing especially on spacecraft ground control and data systems. Although it discusses the evolution of associated technologies, a conclusion is that technology no longer exercises primary dominion over how space operations are conducted. Thus one must also consider the influence and interaction of other factors—especially the sociological and economic factors—to understand and predict future evolution. Parallels are sought in the electric utility, computing, and telecommunications industries. Changes will happen much more slowly than technology will permit or economics will make desirable. We are moving from contributing to the development of technology to being a consumer. Especially relevant to our situation are the concepts of infrastructures and monopolies. When technology obviates the value of services provided by a centralized infrastructure, shifts to decentralization are resisted by human nature, entrenched organizations, regulations and laws.
Keywords: *Technologies, Economics*

History of Technology

The history of technology is a fascinating subject. A macro level examination reveals some characteristics common throughout history—but often not apparent to casual examination. The purpose of this paper is to examine some of these characteristics, then use them to anticipate probable evolution of future space operations.

At its beginning, a given technology is by its nature understood by few—and even those few are grappling with a dynamic learning and understanding process. A corollary is that only these early explorers are able to utilize this newly evolving technology. Thus these specialists become the sole practitioners. But if the technology spawns wide applications, the demands for these practitioners increase, their numbers grow, and understanding becomes more widespread. Other factors then come into play, such as economics and standardization. Entrepreneurs see market potential, and derive new or more efficient products and processes by employing the technology. Efforts at standardization are caught up in competitive battlegrounds. Finally, utilization becomes general and distributed.

Thus a technology evolves from an era of a few specialists proficient in utilizing it, through one of the process becoming sufficiently standardized and automated that needed skill levels drop—and eventually to the point that users unskilled in this particular technology can easily use it themselves to accomplish their individual objectives and desires. It begins at a few scattered points, evolves into a small number of centralized infrastructures, and finally is manifested as common tools distributed to users.

Consider some examples. Automobiles were once built by craftsmen. Each car was an individual product. However, after Henry Ford introduced the concept of interchangeable parts, the services of craftsmen were no longer required. Relatively low skilled people, in far less time, were able to build a car by simply assembling these standardized parts. Or consider commercial airplanes. In an era which some of us can remember, the cockpit crew normally consisted of three people: the pilot, the co-pilot, and a flight engineer. The complexity of operating the aircraft systems was sufficient that a full-time specialist, the flight engineer, was needed simply to operate and monitor them. But as the relevant technologies advanced, these systems became so reliable and automated that the pilot and co-pilot were quite able to operate them in addition to their regular duties. To my knowledge, no modern commercial airplanes now have a station in the cockpit for a flight engineer. However, just as railroad firemen did not disappear when their jobs were made obsolete by diesel locomotives, flight engineers did not disappear quietly.

Among the lessons of history of course is the reality that understanding history certainly does not assure accuracy in forecasting the future, only some understanding of possibilities. It has been said that nothing is so difficult to forecast as the future! Consider in retrospect comments of some well-known leaders in technology[1].

IBM's chairman Tomas Watson's widely quoted observation in 1943 was that "there is a world market for maybe five computers." IBM viewed "big iron" mainframes as ends in themselves, rather than simply as tools to facilitate performing some needed or desired function. Thus they saw the computer market only as large institutional users, and were extremely slow to move into the PC market. An interesting book, Computer Wars[2], describes how this myopia almost became the downfall of IBM.

Closely aligned with IBM were many of the U.S. space programs, which were also very reluctant to abandon Big Iron.

In the same vein, Ken Olsen, chairman and founder of Digital Equipment Corp., said in 1977 "There is no reason anyone would want a computer in their home." Western Union, the telegraph company, said in an internal memo in 1876: "The 'telephone' has too many shortcomings to be seriously considered as a means of communication. The device is inherently of no value to us."

Quoting from a letter a governor of New York sent to the President of the United States at the beginning of the railroad era: "The canal system in this county is being threatened by a new form of transportation known as "railroads."....As you may well know, Mr. President, "railroad" carriages are pulled at enormous speed of 15 miles per hour by "engines" which, in addition to endangering life and limb of passengers, roar and snort their way through the countryside, setting fire to crops, scaring livestock and frightening women and children. The Almighty certainly never intended that people should travel at such breakneck speed." And recall an observation by H. M. Warner, of Warner Brothers movie studios, in 1927 as technology for talking movies became feasible: "Who the hell wants to hear actors talk?" Back to the space program, a 1921 New York Times editorial reflecting on Robert Goddard's work on rocketry said: "Professor Goddard does not know the relation between action and reaction and the need to have something better than a vacuum against which to react. He seems to lack the basic knowledge ladled out daily in high schools."

And then there was Charles H. Duell's famous comment when he was Commissioner of the U.S. Patent Office just 100 years ago: "Everything that can be invented has been invented." He advocated abolishing the U.S. Patent Office.

Or consider the generally acknowledged guru of computers, Bill Gates. Gates was so fixated on putting the power of a mainframe on ever desk that he initially missed the development of the Internet and the power of interconnecting computers using current telecommunications technology. Incidentally, Bill Gates also said—as recently as 1981: "640K ought to be enough for anybody."

Humorous as we may find these remarks with the aid of 20/20 hindsight, when one considers who made them they well illustrate the impossibility of anticipating how technology will be accepted and applied.

What factors really do determine the flow of technology into society? Historically, economics eventually wins. Things which can be done cheaper, ultimately will be. The question is when. Five hundred years ago, Machiavelli[3] wrote: "It must be remembered that there is nothing more difficult to plan, more doubtful of success, nor more dangerous to manage than the creation of a new system. For the initiator has the enmity of all who would profit from preservation of the old institutions and merely lukewarm defenders in those who would gain by the new ones." More recently, Max Planck[4] observed: "An important scientific innovation rarely makes its way by gradually winning over and converting its opponents; it rarely happens that Saul becomes Paul. What does happen is that its opponents gradually die out and that the growing generation is familiarized with the idea from the beginning." While these are sobering indictments of human nature, they reflect the reality that change is difficult for us. To look at the positive side, certainly many changes are accepted—and even enjoyed—in much less than a generation.

Let's recap for a minute the steps in this evolution:

- A new technology is usually developed of necessity to meet some relatively narrow objective.
- The developers are usually actively involved in the application, due to the complex and temperamental nature of the utilization.
- Wider applications are recognized, and a market develops—facilitating more robust and user-friendly concepts and applications.
- Utility expands and prices drop—a very nice "vicious cycle."
- However, vested interests, traditionalists, and especially social and jobs issues, resist this evolution.

Electric Power Infrastructure

In seeking historic parallels to learn from and perhaps help us see into the future, let's consider the largest industry in the United States[5] . Maybe to your surprise, this industry is the electric power industry, with assets of more than $600-billion and annual sales of more than $200 billion. This industry includes a mix of public and private entities competing directly with each other, of technology languishing because of public policies and regulations, and ultimately questions of whether technology has made the concepts of infrastructure and monopoly obsolete in this situation.

Six decades ago the federal government in the United States sought to make electric power available to all citizens as a right, regardless of the economic viability. Few would argue today that this was not a

positive move. However, it is in looking at what has now evolved sixty years later that we can perhaps gain some insight into the future of space operations. Today the federal government is the largest single generator of electricity in the country—supplying almost 10 percent of U.S. power. Supporters of this government industry continually use political clout to prevent taxpayer funds from even being spent to study the amount of subsidization of this industry by taxpayers. However, several independent estimates have placed the value of this subsidy at over $1-billion annually. This situation could suggest that once a government develops the capability to provide a service, the infrastructure behind that service takes on a life of its own. This public industry then strongly resists shifting the service into the private sector even after commercial viability has been established and the private sector dominates the industry.

A second lesson to ponder is associated with the development of relevant technologies. In this case, small power generators have been developed which are generally—on a comparable basis—cheaper to buy, deliver electricity more efficiently, and create less pollution than the old, massive central power stations. Munson and Kaarsberg, writing in *Issues In Science And Technology*, note that while the efficiencies of large natural gas fired turbines have increased from twenty percent to sixty percent over the past twenty-five years, smaller industrial turbines have efficiencies above eighty-five percent when the waste heat is used to produce steam for heating and industrial purposes. They also note Americans now operate over 100-million highly reliable, self-contained engines capable of producing about 100-kilowatts each, at a per-kilowatt capital cost less than one-tenth that of a large electric generator. Where are these 100-million reliable engines? In automobiles. The chairman of one major electric power company predicts that within a decade it will be possible for a 7-Eleven store to install a small package that uses natural gas to produce heating, cooling and electricity. Munson and Kaarsberg do not envision isolated generating units, but rather widely distributed generation, with the power grid remaining in place for load-leveling and backup power. Much of the benefit they see derives from the ability to effectively use the waste heat locally, thus significantly surpassing central plants in overall efficiency.

Whatever the merits of this case, the aspect of interest here is a look at the success of efforts to take advantage of advances in this technology. They cite the case of a major university's efforts to generate their own power as a cost-saving measure. The university expected to save $5.4-million a year, even while planning to pay the local public power utility $1-million a year for standby power. They expected to recover their investment in less than 7 years. First they encountered environmental obstacles. After performing sophisticated life-cycle studies that showed its innovative system had lower net emissions than the state approved technology, they overcame this hurdle. They then became the first self-supplier to be hit with a "stranded cost" charge—an additional charge of $1.3-million a year by the local power utility simply because the utility's investments in generating capacity had been "stranded." The university took this case to court, and eventually won.

The authors' point in reviewing this case—and of interest to us—is that very few have the resources or expertise of an MIT, the university involved, needed to take advantage of technology to save substantial money. They note that part of this specific problem is perceptual: most of us have grown up with the perception that electricity is best produced by distant generators. Thus few question the traditional system, wherein centralized plants discard most of their heat while fuel is then burned elsewhere to produce thermal energy. And certainly a major contributing factor is regulatory and tax policies which do not motivate power companies to incorporate the latest technologies. Again, the availability of enabling technologies, combined with a clear economic case, are not sufficient for the adoption of these technologies. Yet there is evidence deregulation saves consumers. In the four years since Australia began its utility deregulation, wholesale prices have dropped by one-third in real terms

History of Computers

The technology of computers provides an appropriate parallel for reckoning the future directions of space operations. This technology is closely coupled with space operations in a synergistic relationship, with the needs of space programs having influenced the evolution of computing technologies.

The first computers were room-sized machines—complex and cantankerous, requiring much tender care from scientists and technicians who comprehended their idiosyncrasies. Most were devoted to specialized applications, and were essentially institutional assets serving a subset of similar needs of various customers. Two characteristics especially marked their evolution. First, power increased dramatically, following Moore's Law that performance doubles every 18 months. Secondly, facilitated by this increased performance, operation became far more "user friendly."

With this evolution computers gradually moved from expensive, temperamental assets used for specialized applications to commodity tools with wide general value. In a closely related change, highly trained specialists were no longer required to operate these devices. This latter change resulted from at least three factors: reliability improved greatly—especially as millions of hardware connections became

imbedded in silicone, standardization allowed expertise to migrate across systems, and systems became powerful enough for much of the operation to become automated behind simple icons.

Other factors manifested themselves, the dominant one being economics. No longer was the focus on achieving maximum use of a few machines, with the inefficiencies of scheduling, sharing, cueing and waiting being insignificant relative to the capital and operating cost of the system. Rather, this is reversing. Many of us today have two or three personal computers: one at the office, another at home, and a third in our briefcase. The benefits we see in having this tool—a computer—at our disposal anytime outweighs the investment in purchasing more than a single computer at today's prices.

Another significant factor has been the integration of multiple applications within a software suite—a change enabled by the increases in power. Many software packages, widely available at very competitive prices, allow us to move effortlessly among word processing, spreadsheet and database applications, not to mention facilitating integrating the results into attractive presentations. Underlying much of this is standardization—minimizing training and facilitating moving across systems and applications.

Standardization has been down an especially rugged, but typical, road. An interesting example from Munson and Kaarsberg's paper is the observation that in the late 19th century more than 20 different electrical systems operated in Philadelphia alone. A customer moving across the street often found his electrical appliances no longer compatible. Early in the development of a technology there is little reason or motivation to standardize. However, as the technology advances standardization becomes a competitive weapon—with factors such as marketing and politics almost always dominating performance and economics. Recall that in VCR's, Beta technology is superior to VHS, and in production quantities prices were competitive. Yet VHS won the consumer market, and only professionals use Beta today.

Looking at the evolution of computers, note some less obvious changes:

- In the beginning the few machines that existed were basically institutional assets. Only large institutions had the necessary resources and were not driven by need for short-term economic gain. These machines required a lot of tender loving care, and commonly were maintained and operated by their developers.
- As utility grew and as economic factors came into play, consolidation usually occurred within some logical boundaries. Thus these expensive and valuable machines took on the characteristics of an infrastructure—an underlying foundation necessary to accomplish the purposes of an organization.
- Then as the technology matured it provided affordable tools directly usable by those needing to apply it. An often subtle but very important change has occurred at this stage: the tools have been replicated and distributed to customers, and the infrastructure has disappeared!

Where do we see this industry going? There are presently two divergent schools of thought. The Gates faction sees tremendous computing and storage power self-contained on everyone's desktop. The opposing view is one of smaller computers—terminals connected by wideband communications lines with distributed servers containing applications as well as information libraries. While technology will allow movement in either direction, the world seems to be embracing the latter—as demonstrated by the extremely rapid growth of the Internet. An advantage is that it is far easier to update a few servers than many, many times that number of desktop computers.

Telecommunications

Recent developments in telecommunications have been essential, especially the greatly increased bandwidth coupled with decreasing costs. In fact, one company in the U.S. is advertising free long distance telephone service. The "catch" is that users must first listen to a recorded sales pitch before their call is connected.

In decades past, the telephone wires interconnecting homes and businesses within a country were considered infrastructure. With the movement towards deregulation of this industry in the United States, multiple carriers competed for revenues by offering trunk, or long distance, connectivity. However, even then local distribution systems were still viewed as infrastructure, and often treated and regulated as monopolies.

Enter next the cable television industry—which wired communities with coaxial cables, cables able to carry much more traffic than the telephone industry's twisted pair circuits. At this stage we saw two parallel "infrastructures," kept separate by regulation and policy, but technically capable of competing with each other. Then came the wireless telephone industry and cellular phones—initially using local radio cells interconnected with the public switched telephone network, and soon able to link directly to satellites.

At this stage these local telephone wires no longer meet the basic definition of an infrastructure: a "necessary" foundation. Questions of the need to protect them as regulated monopolies arise.

Alternatives exist, and are often more economical. In response, providers of these local telephone wire distribution systems battle with their new competitors in legal and regulatory arenas. In often counterproductive moves, owners of this "infrastructure" sometimes erect toll gates to collect revenues from all who pass—myopic to the reality that by-passing their toll gates is easy, and is encouraged by their competitors. Business travelers can often be seen using their cell phones in hotels to avoid the telephone surcharges imposed by the hotels. An important point to recognize: the basic issues are totally economic and social—business and jobs—and not technology.

The evolution of both national and international telecommunications elements will result in generally favorable changes to space operations. These technologies are certainly being driven by the commercial sector, and are offering continually higher performance—in bandwidth, latency, and bit error rates—at decreasing costs. The changes we will see in space operations are ever more integration of functions and less concern about using distributed ground assets. Without suggesting that today's Internet has sufficient reliability for conducting all space operations, I would point to it as a model of a distributed network of systems for conducting future space operations. For example, orbit determination may be computed on a distant workstation, or in a software module on the controller's desk. Current telecommunications technology will permit the remote workstation approach to perform as satisfactorily as an on-site concept. Decisions between such concepts will be made on other bases, and results will generally be invisible to operators. Just as a pilot on a commercial airplane now has all the information and control needed to operate the aircraft and no longer needs coordinate with a flight engineer, the information flow to and from spacecraft controllers need not pass through intermediaries. The simplest approaches are usually both the most reliable and the cheapest.

Relation with Space Operations

Why have I spent so much time on issues such as electric utilities, the development of the computer industry, and telephone services? Because it is here I see not only technological parallels with space operations, but parallels with the social and economic factors which ultimately dominate and determine the directions and rate of progress—not of technology itself, but of the acceptance and application of technology.

A fundamental truth to me is that advances in space operations today are not significantly limited by technology, but rather by these social and economic factors. Thus to understand the directions in which space operations are headed, our primary attention must be focused on the economic considerations, with special attention to the social factors with which they must interact. We need pay only secondary attention to forecasts of the speed and direction of technology, as these are seldom limiting factors.

How do we apply this notion? The primary consideration must be on economics, and behind economics an understanding of which assets should be shared and which can economically be dedicated. In practical terms, I suggest basically everything from the RF antenna through ground operations and data processing and delivery can be most economically performed today with dedicated systems. Interconnecting telecommunications are readily available at competitive—and decreasing—prices. I base this conclusion on the fact essentially all these functions can be performed with the same computing technology I earlier used for an example—the fact that the adequate performance is readily available at very competitive costs. Certainly specialized applications software is needed, but small companies around the world are actively competing today for this market.

RF Systems

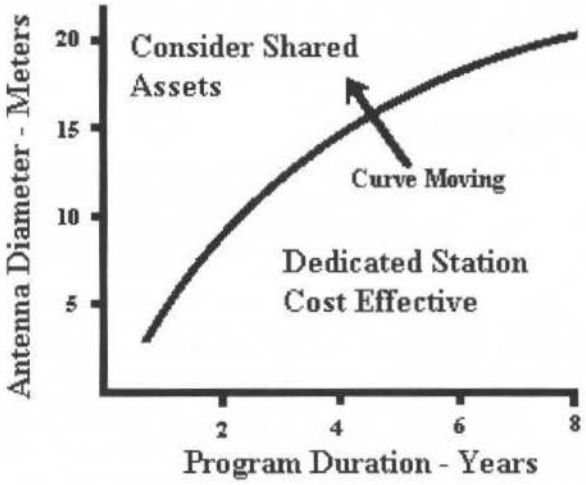
Space operations necessarily involve RF links (including optical links). Unlike the sweeping changes in computing and telecommunications, these technologies are not advancing as rapidly. Antennas, including their supporting infrastructure and maintenance and operating resources, are often high capital assets. Thus sharing often leads to significant cost savings. Here the answers become mission and requirements specific. Spacecraft technology is a fundamental consideration because of the direct interaction with ground systems. One significant development is that of robust, automated spacecraft—spacecraft which do not require constant attention, and are able to either execute corrective actions as needed, or resort to a safe-hold mode pending human intervention. Another is the availability of reliable, high capacity on-board data storage devices. These permit substantial data volumes to be stored on-board until the satellite is in view of a convenient ground station to dump the data.

If the received signals are weak due to extreme distances, low-power spacecraft transmitters, or abnormal spacecraft operation, larger antennas are needed. If small data latency is a requirement, either many ground antennas must be located to permit frequent contacts, or relay satellites must be used. For high inclination orbits, sites in the polar regions will minimumize the number of stations needed, as will

equatorial stations for equatorial low earth orbiters. As orbital altitudes increase, obviously fewer stations are needed. Generally, larger antennas are more expensive, as are data relay satellites. Increasing the numbers of stations needed increases costs, and this increase not linear due to coordination needed within the network. Obviously, the ideal solution from a minimum cost standpoint is for a single ground station with only a small aperture antenna.

However, many satellites are in less convenient orbits, and often complex trade-offs are needed to determine the lowest cost operating concepts. Increasing volume in the antenna markets, along with significant cost reductions in associated electronics, are decreasing antenna system costs—both capital and operating.

Many ground stations today, and all relay satellites, are government owned. Thus prices for obtaining support from these assets are somewhat artificial. However, trading today's prices for buying ground station support against capital plus operating costs for program dedicated stations, yields some practical observations. For almost any program with a minimum duration of a couple of years and without significant latency requirements, a dedicated ground station is usually cheaper if a 10-meter antenna will provide adequate signal margin. A further observation: this cost-effective aperture size is moving up. As either the mission duration increases or adequate signal margins can be obtained with smaller antennas, dedicated stations become more cost-effective. This relationship is illustrated in the Figure below.



Cost Effectiveness of Antennas

Radio Frequency Spectrum

The need for international control of radio spectrum has been long recognized, and the International Telecommunications Union (ITU) was formed for this purpose. Early concerns focused on interference—the unfortunate characteristic of radio waves to totally ignore political boundaries. The coming intense competition for this limited resource was little appreciated. In fact, less than a century ago David Sarnoff's associates gave him this opinion of the future of radio broadcasting: "The wireless music box has no imaginable commercial value. Who would pay for a message sent to nobody in particular?"

With the increasing utility of radio devices becoming obvious, pressures for spectrum have soared. Technology has both greatly accelerated and significantly changed the importance of spectrum in the past one to two decades. For example, telephones once were connected by wires, while televisions received their signals over the airwaves. This situation is now reversing, with wireless telephones common as is cable television. The ITU processes were far too cumbersome to cope with the rapidly evolving technology it was formed to regulate, and it is struggling to adapt. Fights for spectrum are often fierce, and governments see sale of rights as important sources of revenue.

As a heritage of the somewhat casual recognition of spectrum even at the beginning of the space age, we find ourselves not on especially solid ground. Much of the spectrum we inhabit is only under a secondary allocation—meaning we cannot stop primary users from interfering with our operations, while

we are prohibited from interfering with them. As I know from testifying before our Congress on this issue, pressures for spectrum are very strong and there is little sympathy for our dilemma if—or I think more, when—we are denied use of this spectrum.

What will the likely inevitability of vacating much of the spectrum we currently use mean? The precision associated with building radio equipment increases at higher frequencies, a factor which makes the lower part of the spectrum much more attractive commercially. Just as we have already vacated much of the VHF and UHF spectrum, we will find ourselves pushed to continually higher frequencies. The bad news is of course that we will need to replace both spacecraft and ground RF systems in concert—with more costly equipment. Standardization and compatibility will be concerns. However, the technology to move to higher frequencies, primarily Ka-Band, exists today, and the focus will be on reducing costs. The good news is that moving up in the spectrum will enable the much higher data rates which space operators such as earth imaging are demanding. The primary challenge will be in making the transition because of the inter-relation among so many systems, in space, on the drawing boards, and on the ground. In a synergistic way, space operations have contributed to the development of technologies such as coding, enabling much more efficient use of limited spectrum.

At the top of the spectrum possibilities is, of course, optical. I have no doubt optical space-ground communications will be in our future. It not only offers essentially unlimited bandwidth, but will also reduce the capital and operating costs of the ground collectors. The obvious problem with clouds can be solved with a sufficient number of ground stations to achieve an acceptable statistical probability of a satisfactory link margin. Imagine the economics of replacing large radio antennas with small optical telescopes. This is one place the technology only marginally exists today, primarily in the area of high power diode transmitters for the spacecraft, and secondarily in pointing accuracy and stability. However, there is little doubt this technology will be mature long before we are able to incorporate it into space systems. Remember also that most of these technologies are already being driven today by the commercial fiber industry.

To summarize this aspect, I anticipate space operations will be driven to ever higher radio frequencies. This move will be a dominant factor in planning new systems, both in space and on the ground. Spectrum regulation and competition will be the driving factors, resisted by the costs of changing space and ground systems, and by the difficulties of coordinating the changes among multiple operational elements. Technology will be available and not a big consideration. Ultimately the impacts will be the economics of system cost, and little directly perturb space operations.

Other Factors

I have only alluded in passing to the one issue I feel is dominant: the jobs issue. The reason is that I frankly don't know what to say. I have learned the hard way that it is an old and significant reality, and certainly an understandable one if we imagine our own livelihoods being threatened. We ignore it at our peril. One observation: effective resistance is usually organized—unions and employers. This resistance often reflects involvement and genuine concern for those adversely affected. However, it is also true that these motivations sometimes are more directed to preservation of the organization itself than to its people.

In addition to spectrum and telecommunications, other regulatory policies are of some concern, particularly environmental regulations. These focus mostly around the radiation hazards of high-power transmitters, and sometimes the aesthetics of large antennas.

Summary

If you accept my hypothesis that neither technology nor favorable economics will be determinant in forecasting the future of space operations, what factors will be influential, and how might they evolve?

The primary factor is our human reluctance to change ways that have served us well in the past. This resistance is amplified by the subsequent entrenchment of organizations, regulations, and laws protecting and preserving the status quo. Technology is only a factor insofar as it is limiting, and thus guiding, the development of space operations—generally this is not the case. Driven by commercial motivations, development of most relevant technologies is now outpacing our ability to capitalize on them. Thus we will normally have the opportunity to select proven technologies as socio-economic conditions allow us to move forward.

Especially relevant to our situation are the concepts and roles of infrastructures and monopolies. Infrastructures are usually either public, or private but operated under regulatory protection. When technology obviates the appropriateness or need for services to be provided by a centralized infrastructure, we confront the existing entrenched entities. Transitions from government to private

sector suppliers, or deregulation of regulated monoploies, are usually very rocky, with the "losers" resisting any change and the potential "winners" fighting for windfalls from the transition.

My crystal ball is no better than any of yours, and I cannot provide any definitive forecast of the future of space operations. However, for those like me—engineers who prefer predictable things like Ohm's Law, and who are enchanted by trying to predict technology—it is essential that we understand the true nature of the forces confronting us.

The fact that something is new, the fact it is possible, the fact that it is cheaper, certainly are not of themselves reasons to embrace it. But neither is the fact that something is new sufficient to ignore evidence it is technologically feasible and economically superior. The future direction of space operations will be toward simplicity, toward an era where operations will become an incidental part of our life and work, much as we operate our automobiles with little thought to the mechanics—and sometimes with not much more conscious thought to traffic and road conditions! Unpredictable is the rate of acceptance. This will be determined by economic and political imperatives, not by technology. History suggests an outer limit for this time: the time we disappear from the scene and a new generation takes over! Hopefully we will leave them a better heritage.

References

Science and Technology Quotes http://busboy.sped.ukans.edu/~adams/sciquot.htm

Fergunson, C. H. and Morris, C. R., 1994, "Computer Wars: The fall of IBM and the future of global technology", Times Books.

The Prince

The Philosophy of Physics, 1936

Munson and Kaarsberg, 1998, "Unleashing Innovation in Electricity Generation", Issues in Science and Technology Online, Spring.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp.260-266

ISSN 0100-7386
Printed in Brazil

Tools for Operations Preparation and Automation: the Opsware Approach

François Lecouat
Jean-Michel Darroy
Matra Marconi Space
31, Avenue des Cosmonautes
31402 Toulouse, France
francois.lecouat@tls.mms.fr

Abstract

The automation of ground operations is a key issue for reducing exploitation costs of satellite systems while improving safety and the availability of the provided service.
Many operations managers are considering the implementation of new functions for automating operations. Their motivations can be separated in two classes:

- *Minimize the staff or operate more satellites with the same staff, and thus reduce exploitation costs. For this purpose, automated operations allow an optimized off-line preparation of operations by experts during working hours, and they relieve operators from routine or time-consuming tasks (detailed monitoring of parameters, preparation and release of commands, ground station management)*
- *Improve safety of operations and availability of the space system. For this purpose, automated operations reduce the risk of human errors (a common cause of service interruption) thanks to a complete, rigorous, repetitive and fast execution of planned operations. Moreover they allow a larger number of verifications, a reduced reaction time upon anomalies and a dynamic optimization given the available ground and space resources.*

To address these issues MATRA MARCONI SPACE has developed a set of scalable complementary products based on state of the art technologies, called OPSWARE. This paper provides users feedback on these flight proven products, and describes the latest improvements of these tools that are continuously updated to take into account experience from on-going missions.

New Functions Required for Operations Automation

A traditional control center generally provides the following functions:

- Telemetry processing and display
- Command preparation and release
- System Database preparation and browsing
- Flight dynamics

Sometimes a scripting language is available to program command sequences that implement a first step towards automation. However these sequences are often difficult to prepare, validate, operate and maintain, and are thus only used on a small scale.

In order to reach large scale automation objectives, new functions are being progressively implemented in existing and in new control centers:

- Procedures preparation and execution
- Plan preparation and execution
- Data analysis and report generation

Various efforts are also made to support diagnosis and fault isolation. Up to now general approaches, based on the expert system and diagnosis theories, proved to be too costly in term of satellite and operations model preparation (low return on investment), so the problem is being addressed with small improvements of the Telemetry Processing or the Data Analysis functions.

Traditional Functions	Telemetry Processing	Commanding	System Database	Flight Dynamics
New Functions for Operations Automation	Procedures Preparation and Execution	Plan Preparation and Execution	Data Analysis and Report Generation	

Procedures Preparation and Execution

All operations of a spacecraft are, in principle, completely defined by a large manual of operations procedures known as the Flight Operations Plan (FOP). This comprises Flight Control Procedures covering nominal operations, and Contingency Recovery Procedures which describe the actions to be

taken in non-nominal situations. This concept of procedure is also often used for controlling and commanding the ground segment (mission control center, network, and ground stations).

Since procedure books are the source of most of the decisions and actions made by operators, several attempts have been made to place procedures at the heart of automation, through a computerization of procedures. They are stored in a representation that can be understood by a machine for execution and that can be printed to form procedure manuals that are supposed to replace traditional manuals. A procedure preparation function shall permit to enter procedures that can be passed to a procedure execution function.

This shall lead to a global approach that avoids the maintenance of traditional procedure manuals and a large collection of command scripts that partially implement procedures.

Plan Preparation and Execution

Within a control center, the activities required to maintain the satellites and the ground segment in an appropriate state are planned by operations engineers, and form the operations plan.

Engineers select required activities (including spacecraft control procedures, ground system control procedures and ranging actions), define their parameters and define the time at which they are to be performed. They must respect precedence constraints between activities (activity A must be done before activity B), resource constraints (Activity B and C require the same antenna and cannot be done at the same time), and priority constraints (activity B is more important than activity C).

Some planning functions are being developed to facilitate this planning task for launch or routine operations, and the replanning that may be required if the assumptions change. The objective is to produce a global view of the operations plan (with a representation of the constraints) and a detailed view with the dated commands of all the procedures to be executed (detailed timeline document).

Another planning problem consists in preparing payload activities (images to be taken, telecom repeater configurations) that maximize the use of space resources and respect constraints (power, bandwidth, priorities). This problem is often addressed with an ad-hoc planning function that generates a payload plan that is translated into a set of control procedures to activate or reconfigure the payload. So the payload plan can be seen as a part of the operations plan. The later is more general since it also addresses satellite platform operations and ground segment operations.

Once an operations plan has been prepared with a planning function it can be tempting to go further and implement a plan execution function that automatically initiates the appropriate activities when their execution time comes, in order to achieve global automation. A few attempts are being made in that direction.

Data Analysis and Report Generation

Within a control center, the operations personnel has to monitor the performances of the controlled satellites (platforms and payloads), to detect trends in order to prevent or to predict anomalies. The output of this task is a set of reports that are produced on a daily, weekly or monthly basis.

Usually the Telemetry processing function of the control center supports this task with features to retrieve, visualize, and plot recorded parameter values.

These features are often insufficient for sophisticated trends analysis, and additional functions are being developed to implement powerful retrieval and vizualisation functions, math operators, production of report templates, and automation of analyses in batch mode.

Opsware Approach to Operations Automation

To meet the requirements of operations automation described above, MATRA MARCONI SPACE (MMS) has developed an operations concept and the associated line of products called OSPWARE.

Origin and Scope of the OPSWARE Products

MMS is involved in the development of all kinds of space systems (telecommunication, observation, and science missions, launchers and manned flights) at prime contractor or sub contractor level. In the area of ground segments, MMS is providing complete solutions and sub systems (ground stations, control centers, mission centers, simulators, etc).

The OPSWARE products are the result of a continuous effort that started in 1985 to develop software for operations preparation and execution, to optimize MMS internal processes (operations preparation, integration and validation, spacecraft control, in-orbit follow-up), and to provide high value software products for MMS customers.

The foundation for OPSWARE is a unique experience in spacecraft operations coupled with strong capacities in computer science. Over the years this effort has produced prototypes, then operational

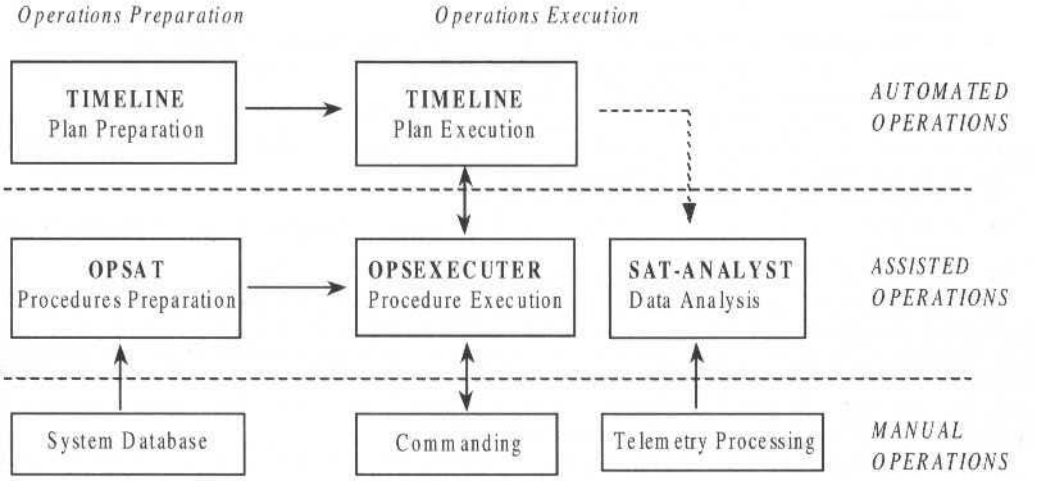
applications and finally products. The second generation of products is flight proven, and a third generation is now being released.

The OPSWARE products address all the operations preparation and execution tasks described in §1:

- Procedures Preparation and Execution (OPSAT and OPSEXECUTER)
- Plan Preparation and Execution (TIMELINE)
- Data Analysis and Report Generation (SAT-ANALYST)

These products are marketed as part of complete solutions for command and control provided by MMS or as add-on functions on top of existing command and control software. They have been designed to be:

- Suited for all kinds of missions in GEO and LEO orbits, from simple missions to large constellations,
- Independent from any spacecraft design and thus usable to operate spacecraft from any manufacturer,
- Independent from each other, to allow progressive deployment,
- Easily interfaced to existing infrastructures



Typical architecture for operations automation based on OPSWARE. The first layer on top of telemetry processing and commanding functions provides assisted operations based on automatic procedures execution. The second layer implements automated operations based on plan execution.

Open Architectures Based on Advanced Technologies

Most operations automation functions are part of the critical control command chain, and thus require a high reliability. Moreover these functions must achieve high level of performances. Finally considering the long duration of space projects (up to 15 years) these functions have to be maintained for very long periods of time. OPSWARE products are based on the most advanced software technologies that are available today for the required reliability, performances, and maintainability, such as C++, ORACLE and CORBA. Proprietary components are avoided or used in well isolated modules in order to facilitate evolutions and increase life duration thanks to the use of standards.

Products are coded with rules for high quality and portability. They are available on several operating systems and can be easily ported to new operating systems.

Each product provides an extended and well designed Application Programmer's Interface (API) in order to facilitate:

- Connection of the product to existing infrastructures (system database, command and control layer),
- Customization of the product to the local operations concept,
- Extension of the product with new functions.

A client-server architecture supports the deployment of completely distributed solutions based on CORBA, that allow all the authorized users to get a specific view on operations.

Procedures Preparation and Execution

Version 3 of the OPSAT and OPSEXECUTER products for procedures preparation and execution have been released at the end of 1998.

The functions provided by OPSAT are :

- A specialized editor which supports "assisted editing" in a formal language that supports a syntax close to the natural language of users, and a structured layout to increase readability. It provides on -line access to the system data (TM, TC, ...) and numerous on-line helps (syntax-driven editing, ...) for more efficient procedures writing.
- A database to save and distribute procedures that enables multi-user access and also the implementation of powerful functions for searching impacts of modifications of the system data upon the procedures (e.g. find all procedures using a given TC). The tool allows to manage several flight models: the user can display differences between several models, and export data between models. This function can also be used to exchanges procedures between several teams working on a given spacecraft.
- A procedures compiler, which generates an internal, object representation of the procedures, detects syntactic errors and verifies the consistency of procedures with respect to the system database.
- A procedures formatter, which automatically generates high-quality procedure documents. The formatter creates a command file for a standard desktop publishing tool. A single procedure or a complete manual can be formatted at a time.

Procedures prepared with OPSAT can be exported to OPSEXECUTER for execution. There is no need to recode them in a computer representation that may be obscure to operators: OPSEXECUTER is acting on the validated procedures of the Flight Operations Plan.

For each procedure two execution modes are provided :

- A manual mode in which the user must acknowledge the execution of each instruction of the procedure;
- An automatic mode in which the procedure is executed without interruption.

In both modes, the procedure can be shown on the user interface with the current instruction highlighted, and all information acquired during the execution (TM values, TC pre-execution checks and post execution verifications, ...) are automatically reported in this display in front of the corresponding instruction : a good observability of the process is thus ensured. The automatic generation of an "as-run" procedure document, showing a detailed trace of the execution in front of each procedure instruction is provided. These files can be saved to build-up a structured history.

The user can get back in the loop at any time, in order to return to manual mode, skip or abort the current instruction, stop the procedure, send an individual command.

For each kind of procedure instruction the tool takes care of contingency management (delays, bad TM, failed commands...). This allows to limit the number of instructions written by users.

OPSEXECUTER also provides high level views on the procedures under execution that gives a synthetic report on the execution status.

Plan Preparation and Execution

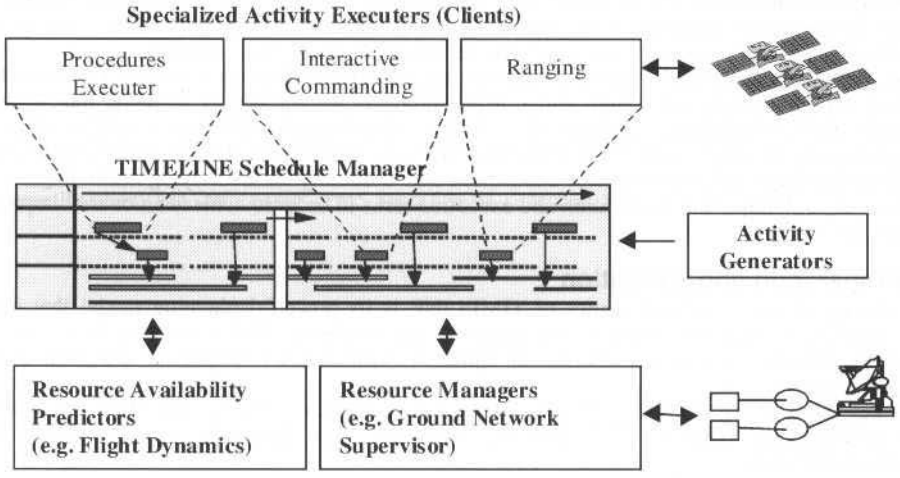
Version 3 of the TIMELINE product for operations scheduling and plan execution has been released at the end of 1998. This version provides a new architecture in order to increase the genericity and the scalability of the product.

TIMELINE is a planning server that permits to build a plan interactively from the graphical user interface or from requests from other software applications that can be connected to the server through a well defined client API.

TIMELINE can schedule the activities of the plan: it assigns start dates to activities that respect the various constraints (if possible) and allocates resources to activities. The tool is designed for large problems: it can handle thousands of activities.

TIMELINE provides extensive plan execution capabilities. It can activate resources through dialogues with resource managers such as a network supervisor. It can start activities at the appropriate date by sending request to specialized activity executers such as OPSEXECUTER, and can monitor the execution of the activities through dialogues with the executers. It is able to react to various kinds of events and make decisions that are translated into messages to activity executers and resource managers, and may force a rescheduling of activities.

TIMELINE provides a powerful graphical user interface that can present several views of the plan (in Gantt or tabular forms) and can be easily customized.



Thanks to its plan execution capabilities TIMELINE can provide an unprecedented level of automation and act as the orchestra conductor of the whole ground segment. The product can manage few satellites up to a large constellation, in LEO or GEO orbits.

The product can also be used for off-line mission planning, for instance to assist the preparation of the launch phase of a spacecraft, or for payload activities planning.

Data Analysis and Report Generation

SAT-ANALYST is a generic product for data analysis and report generation. It is a ready to use tool that requires no programming skills and can be used as is by operators.

It provides powerful graphical facilities (plots and data tables) and numerical facilities that can be used interactively or within recorded analysis procedures that can run in batch mode. The tool can generate report documents or plots in various formats. It can handle vast amounts of data that make collapse most plotting utilities, and it manages telemetry holes (no value available) in a clean way.

Version 4 of the product has been released at the end of 1998. This version provides functions that allow to build and manage statistics (min, max, average) that permit to perform long term trends analysis along the lifetime of the spacecraft. Other new functions include table management, data access interactive configuration, and a richer set of graphic formats for insertion in Word documents.

User Feedback

The OPSWARE products are heavily used at MMS, on various kinds of programmes in telecommunication, observation and science, for operations engineering, spacecraft integration and test, spacecraft control, and in-orbit follow-up.

They are marketed by MMS, and are now being deployed at customers sites on all continents on programmes such as NILESAT, ST-1, ASTRA 2B, WORLDSTAR, HISPASAT, SPOT, etc. OPSWARE tools have been selected by INTELSAT to implement its new ground infrastructure within the Flight Dynamics and Commanding system (FDC) that will permit to run safe and highly automated operations of the INTELSAT satellites fleet (30 satellites from 7 series of various constructors).

User feedback is essential for MMS, since the company makes a significant effort to continuously capture experience from on-going missions in order to improve the products and the operations concepts supported by the products.

Procedures Preparation and Execution

The generation, validation and maintenance of large high quality procedures manuals is a very costly task. Feedback from various contexts shows that OPSAT allows to divide the preparation tasks by 2 or 3, and reach a quality that is much higher: procedures are normalized, no time is spent on formating, errors are detected earlier without costly simulations, impact of changes can be analysed quickly, procedure parts can be reused, and satellite series are managed.

Feedback on OPSEXECUTER is also excellent. The tool is handled very easily by operators who quickly build a high level of confidence, and make it one the central function of the control center, even in critical phases such as launch operations. This is attributed to the model provided to the operator that is

quite intuitive and to the good observability and commandability of the product that allows to get back in the loop at any time. Operations managers also appreciate that a lot of checks are performed systematically, and that procedures execution can be very rigorous and yet very fast.

Procedures preparation and execution functions are at the heart of the processes of the control centre. They have a high criticity and a high value. MMS products are the results of more than a decade of experience. Several procedure concepts have been experimented including a graphical procedure representation (enabled by an expert system shells that is now outdated). The main lessons was that it is essential to consider the performances, robustness and durability of underlying technologies, and to consider as equally important preparation and execution tasks in order to achieve an overall improvement of operations.

Plan Preparation and Execution

According to users the first quality of TIMELINE is its powerful planning model that allows to handle a very large répertoire of planning scenarios and strategies. This is coupled to a very flexible graphical user interface that allows to represent a plan in a customized way specific to a given mission, or a class of users.

The key point is that a TIMELINE plan is not a rigid sequence of activities. The planning model permits to explicitely represent the constraints of a plan. This allows the scheduling and the plan execution modules to:

- Optimize activities given available resources
- Build robust plans that can resist to small anomalies
- Quickly replan when assumptions change or unexpected events happen

Preparing plans in advance with explicit constraint representation introduces more exactness and enables optimization and analysis of back up scenarios.

The rich interfaces allow to connect the product to many software applications (activity generators, specialized executers, resource availability predictors, resource managers) and thus allow an unprecedented level of automation.

The tool provides a good situation awareness (global or detailed views of past, current and future operations). The status of activities and resources are updated in real time: approbation, validity (logical constraints), conflicts (temporal and resource constraints), execution state.

The user can get back in the loop at any time to modify the plan if necessary or to acknowledge critical operations. Thus the level of automation is adaptable.

Data Analysis and Report Generation

SAT-ANALYST has been working for 3 years at MMS to perform the trends analysis and reporting tasks on more than 10 satellites. It is also used for spacecraft integration. Each day hundreds of graphs and tables are automatically generated and inserted in reports. Thousands of statistics parameters are automatically updated. SAT-ANALYST is also used for investigation and diagnosis purposes particularly during launch periods.

Compared to standard office software that first comes to mind for data analysis tasks, the users highly appreciate the good performances of the tool, the fact that it can be used as is, without programming, and the capability to handle telemetry holes. New statistics functions allow to save a lot of time.

The tool allows to progressively refine the exploitation of data, based on the experience acquired over time, and by this way capitalizes the technical memory of operators and engineers.

Conclusion

Users feedback shows that OPSWARE products embody an operations concept that allows very significant efficiency and safety improvements. It completely addresses preparation tasks instead of only focusing on operations execution. It allows a high degree of automation, that is adaptable in contexts where less automation is required. It permits to react to events and anomalies, and to optimize the performances of the operated system. It builds operator confidence.

References

[1] Darroy, J.M., Lecouat, F., Brenot, J.M., and de Saint Vincent, A., Oct. 1993, "OPSWARE: A new generation of software tools for making space operations faster, better and cheaper", 44th Congress of the International Astronautical Federation, Craz, Austria.

[2] Lecouat, F., Parrod, Y., Pham, P. and McMahon, D., Sept. 1995, "Architectural Concept for Operations Automation", International Symposium on Reducing the Cost of Spacecraft Ground Systems and Operations, RAL, Oxford, UK.

[3] Lecouat, F. and de Saint Vincent, A., Sept. 1996, "System Autonomy through Ground Operations Automation", Space Ops 96, Munich, Germany.

[4] Lecouat, F., March 1997, "Reducing Satellite Operations Costs with the OPSWARE Tools", International Symposium on Computer Tools and Systems Engineering, AAAF, Paris.

[5] Gasquet, A., Parrod, Y. and de Saint Vincent, A., Nov. 1994, "Towards a New Generation of Mission Planning Systems: Flexibility and Performance", Proc. Space Ops 94, Greenbelt, MD, USA, Nov. 1994.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp.267-274

ISSN 0100-7386
Printed in Brazil

Hubble Space Telescope: Transitioning to 1990's Technology for Operations

Julio L. Marius
NASA
Goddard Space Flight Center
Greenbelt, Maryland, USA 20771
jmarius@hst.nasa.gov

Jack E. Leibee
NASA
Goddard Space Flight Center
Greenbelt, Maryland, USA 20771
jleibee@hst.nasa.gov

Manfred Miebach
Space Telescope Science Institute
3700 San Martin Drive
Baltimore, Maryland, USA 21218
Miebach@STSCI.edu

Abstract

A new control center for the Hubble Space Telescope (HST) has been developed and is scheduled to become operational in early 1999. The new control center required a new computer operations system, the HST Control Center System. The control center facilities also required changes to accommodate a different operations concept and new ground system hardware.

Because of the complexity and scientific worth of the HST, a careful approach to transitioning operations from the existing system to the new system had to be developed. The transition has been unusually difficult in that normal operations continued during refurbishment of the facilities.

This paper discusses the management, technical processes, and lessons learned in transitioning complex operations from a heritage system to a new system. It also discusses the new system's performance status. The lessons learned from the transition, and the processes developed for it, may apply to other simple or complex, new or existing missions.

Keywords: *Hubble Space Telescope, Operations, Transitions.*

Background

Although the Hubble Space Telescope (HST) was launched only 8 years ago, its ground control computer system was developed in the late 1970's, a time when mainframe computers still ruled the world. The facility in which the computer system is housed and ground operations have been conducted, the Hubble Space Telescope Operations Control Center (STOCC) at the Goddard Space Flight Center (GSFC), uses custom code and hardware designed and built in the 1980's.

STOCC required approximately one million lines of executable code for its four primary functions:

- spacecraft health and safety monitoring;
- command and control;
- engineering data trending and analysis; and
- sensor analysis and calibration.

Four different contractors built these functions into four distinct computer systems that were operated and controlled by computers in four different rooms, (Gainsborough, J., 1997).

During 28 years of continual patching and modification, the overall ground control system has become increasingly heterogeneous. It has incorporated dozens of computer platforms to accommodate ever-changing spacecraft needs. Operations are intensively manual, and maintenance of the aging hardware and software systems has become increasingly expensive.

In 1994, in light of a gradually shrinking budget for HST for the remaining 20 years of its useful lifetime, the HST Project personnel at GSFC initiated a re-engineering effort known as the "Vision 2000 Project" to streamline and modernize HST operations. Part of the modernization was a new computer Control Center System (CCS) to operate and control HST.

The primary goals, (Gainsborough, J., 1997), for the new CCS were:

- to reduce overall operations and maintenance costs by automating routine functions and using off-the-shelf hardware and software;
- to provide automated monitoring for the ground and spacecraft systems so that operations personal were involved only on exception basis; and

- to provide authorized worldwide access to ground and space data to perform engineering diagnostics using the latest internet technology.

The new CCS incorporates the four primary STOCC functions in a single distributed system that allows any function to be performed from anywhere in the complex. Its architecture allows distributed monitoring of HST operations, including public monitoring, both in the USA and around the world. CCS is comprised of commercial off-the-shelf hardware and software, reused legacy code, and new code using the C++ programming language. It also includes two new HST telemetry repositories, essential systems for engineering monitoring and trending. The CCS facilities themselves were also changed to accommodate the different operations concept and the new ground system hardware.

Main Technical and Management Challenges

Transitioning to 1990's technology for the HST ground system was a challenge from both technical and management standpoints.

The primary technical challenge was to build a command and control center which improved the performance of the legacy ground system but maintained its "feel." This was important because operations personnel had over 8 years of experience with the existing Ports Refurbishing System (PRS). The PRS software was all custom code with some custom hardware. The look and feel of the control center, including CRT displays, consoles, overhead displays, ancillary equipment such as PCs, and even the room layout, all were ingrained in the operations. Thus was appropriate that such extensive and pervasive experience apply to the new ground system.

There were two major management challenges:

- develop and transition to a new control center while maintaining ongoing operations without an increase in operations staff; and
- convince the flight operations team that the new system would be an improvement over the existing system.

The first management challenge was in some ways easier to meet than the second in that the development and transition personnel did not resist changes.

Technical Challenges

Magnitude of HST Operations

It is necessary to understand the magnitude of HST operations in order to understand and appreciate the effort required to transition to a new control center.

HST orbits the earth every 97 minutes at an inclination of 28.5 degrees with a nearly circular orbit of 330 nautical miles. The NASA Tracking and Data Relay Satellite System (TDRSS), comprised of several geosynchronous communications satellites, provides almost continuous command and telemetry communications for HST. Every orbit, there is an approximately 12 minute communications outage when HST cannot communicate with a TDRSS spacecraft.

Thus, since the HST launch, the Flight Operations Teams (FOTs) had monitored HST engineering telemetry for 85 minutes out of every 97 minute orbit, spending most of their 8- or 12-hour shifts sitting at the same consoles and looking at the same surroundings.

FOT Responsibilities

The FOTs are responsible for monitoring engineering telemetry data that is downlinked at a rate of 32 kilobits per second and contains over 6800 telemetry parameters. Although many of the telemetry parameters remain static, there are a considerable number that change continuously and are sampled up to a maximum rate of 40 Hertz per second. It is the responsibility of the ground system computer and the FOTs to detect any erroneous telemetry and react accordingly.

There are three primary methods to monitor HST data.

- The PRS ground system automatically checks for limit violations which are pre-defined and maintained in a database.
- The PRS performs conFiguration monitoring upon operator command and compares existing conditions with pre-defined expected conditions. Any deviations are noted and an alert sent to the operators.
- The third method was simply having the FOT personnel monitor the displays and compare the information against a variety of mission products.

Although this latter method appears old-fashioned and may seem unnecessary, there have been multiple occurrences when it was the only, or the best, or the first way to detect a problem. In one instance, the FOT took preventive action and placed the spacecraft in a safe configuration before the computers in the ground system or the spacecraft even recognized that anything was wrong.

FOTs are also responsible for command functions. Over 125,000 commands are uplinked to the spacecraft every week in order to execute the science program. Due to the limited memory size of the two computers on the spacecraft, loads for one computer are uplinked once a day and loads for the other computer are uplinked three times a day. Failure to successfully uplink a load in the allotted time frame results in the spacecraft entering a protected condition known as safemode.

The other commanding functions are to downlink science data from records on a solid state device, and to uplink ephemeris data to provide spacecraft pointing for target acquisition and for radio frequency communications with TDRSS.

CCS Technical Planning

The technical challenge of building a new HST control center was driven by the need to be able to downsize operations and software maintenance costs. Both types of cost are relatively high when compared to other missions. However, HST was the most complex scientific satellite spacecraft ever built; there are no comparable NASA scientific missions. Comparisons to the Space Shuttle are not relevant due to the manned factor of the Shuttle. What made HST even more difficult to operate at low costs was the requirement to service HST every 3 years while maintaining normal HST operations. The planning and preparation for servicing missions required changes in the ground system to accommodate new spacecraft hardware.

A list of goals and objectives was created by a the Operations Concept Development Team, a group of individuals who had extensive experience in both spacecraft operations and ground systems development. This group did not set forth an architecture or a specific operations concept, but left that up to the System Architecture Team. This second group took the goals and developed a system architecture which would meet not only the functional goals but the cost goals as well. The primary result was a new architecture which separated functions logically rather than along organizational lines.

The original HST ground system architecture and development was heavily influenced by organizational lines of responsibility as opposed to technical considerations. A factor that made the architecture and implementation more difficult was the division of responsibility for HST between the Marshall Space Flight Center in Huntsville, Alabama, and GSFC in Greenbelt, Maryland. Marshall Space Flight Center personnel were responsible for building the spacecraft and conducting the spacecraft commissioning phase, while GSFC personnel were responsible for overall operations and development of the scientific instruments. A further complication was that within GSFC, two different organizations were responsible for building the ground system.

As a result of the divisions of responsibility, the original planning and scheduling system and the original online monitoring system were both divided and duplicated to some extent. This resulted in a ground system which was not only difficult to maintain but also had numerous artificial boundaries between functions which would otherwise be logically grouped.

The new architecture combined the two planning and scheduling systems into one, and also combined multiple engineering data processing, archival, and trending systems into a single system.

The System Architecture Team tasked two separate teams with developing internal architectures, developing the code, and delivering the new systems to over 10 different locations and users. One team was responsible for planning and scheduling; the other team was responsible for real-time operations. Each team was headed by an individual who was selected for competency and experience, regardless of organizational affiliation. Each team was comprised of individuals representing more than 7 different companies plus NASA. Each team determined the best management structure for their own needs and also developed separate processes for building their products.

In addition, three other teams were formed to address two of the spacecraft computers and the science data processing systems. These teams had already existed as other entities, but were reorganized to better fit the paradigm for building a new ground system.

In order to ensure that the two efforts were not totally disparate and to prevent unnecessary or incorrect work from being done, a Systems Engineering Board (SEB) was established. The SEB was chaired by the Chief Architect and was comprised of each of the development team leaders plus key operations and development personnel from the HST Project, including the HST Operations Manager, the HST Operations and Ground Systems System Engineer, and the Vision 2000 Program Manager. The SEB met on a biweekly basis, established overall standards, and approved changes that affected multiple systems. System tradeoffs, mostly involving space to ground, were made by the board.

A measure of the success of the architecture was its recent use by a commercial communications satellite operator, Globalstar, LP, in San Jose, CA, to aid in delivering voice, data, fax and other telecommunications services to users worldwide. The spin-off technology will satisfy the critical need for Globalstar engineers to access spacecraft telemetry data remotely from anywhere in the world. Globalstar team members and partners will be able to coordinate efforts and dynamically monitor and troubleshoot situations with the constellation of 48 low-Earth orbiting satellites from multiple locations.

Management Challenges

One of the biggest challenges was managing the transition from the existing system to the new CCS. As previously noted, the heritage system was ingrained in everyone's psyche, and spacecraft operations couldn't be interrupted.

Development Team

The Development Team was empowered to address all the various activities in developing and executing a transition plan. The team was comprised of representatives from operations, software development, hardware maintenance, management, and systems engineering. Focus teams addressed specific areas of the transition, such as conversion of existing procedures; facility modifications; training; configuration management; CRT display conversion; engineering trending and analysis conversion; and system maintenance and management.

Transition Team

The Transition Team was chaired by the HST Operations Manager. Its charter was to develop and execute a process which would safely transition operations from the legacy system to the CCS without disrupting or introducing additional risk to on-going operations.

Approximately 2 years before transition, the team began meeting weekly to begin laying out a plan and schedule. The myriad of activities required for transition were identified and leadership responsibilities assigned. Thirteen separate groups were formed. The groups included: facilities; training; infrastructure; procedure conversion; display conversion; testing; engineering telemetry analysis verification; and configuration management.

This division into groups worked well until it became apparent that some of the work to be done by the Management Team was not being accomplished on time, primarily due to a greater than expected development effort. When this problem arose, a new set of Transition Team focus groups was formed to replace most of the previous Transition Team subgroups. The new focus teams concentrated on the developmental processes of telemetry; command; system monitoring and configuration; archiving; and procedure translation and training.

Also, the new Transition Team focus groups were relocated to the development facility and given the responsibility to provide a turnkey product,: i.e., a system that could be used operationally by the mission operations team.

This organization resulted in significant improvement. The frequency of problems reported was reduced by 75% within two months. Equally important, the rate of procedure development doubled in just one month.

Telemetry Archive Transition

Functional Architecture

Telemetry archiving is one component of the CCS. The overall CCS system architecture is outlined in Fig. 1: CCS Architecture. Mainly for security reasons, CCS consists of Core and Backbone systems separated by "firewalls" to control the message and data traffic between them.

By design, the Core system was limited to functions that need tight control, such as commanding the spacecraft and handling the NASCOM interface. The Front End Processor (FEP) provides a 2-way communication interface between the spacecraft and the ground systems. The Core system performs all telemetry decommutation, conversion, simple limit checking, and time stamping for real-time as well as recorded telemetry. Command handling was based on inputs from the Planning and Scheduling (PandS) System that provides command loads, orbital data, scheduled timelines, and expected state information of the different subsystems onboard the Hubble Space Telescope.

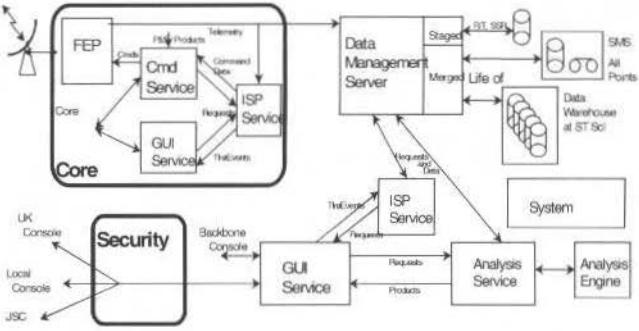


Fig. 1 CCS Architecture

The Backbone system receives real-time and recorded telemetry from the FEP. The Backbone System contains two HST telemetry repositories:

- the All Points Archive, and the
- Data Warehouse (DW).

The All Points Archive contains a complete record of the HST telemetry. All telemetry data received from the spacecraft, and real-time derived mnemonics, are stored here on-line on disks for about 30 days.. The All Points archive provides fast access to the most recent data. The archive is particularly useful to analyze short intervals of high-rate data (e.g., gyro rates). Query capability -- compared to that of the Data Warehouse -- is limited.

The Data Warehouse was designed for more complex query capabilities. It provides telemetry data from the entire lifetime of HST stored on tape or optical secondary storage. This archive system is optimized for fast query access. The data contents are determined by Filter Definition Files which allow the following data manipulations and result in reduced storage volume.

- Selected mnemonics are discarded (e.g., spare monitor points).
- High-rate and some temperature points are averaged.
- The majority of mnemonics are stored as change-only data.

The Data Warehouse also contains a variety of events (orbital events like day/night transitions, changes in instrument configurations, command events, etc.) that can be used as constraints on complex data queries.

System Monitoring realizes is the most advanced CCS function provided by in the area of spacecraft and ground system monitoring. The system performs real-time monitoring of spacecraft telemetry based on expected state and configuration information derived from the command stream. With this, the system can detect and analyze anomalies and automatically alert mission engineers. CCS events are logged and filtered to immediate attention.

The Graphical User Interface (GUI) provides a common interface between CCS functions and data, including access to the telemetry repositories. In particular the GUI system will allow the user to:

- build, save, and recall real-time display pages customized to specific subsystems; and
- use display pages to access historical data, perform trending and analysis, commanding and system management functions.

The implementation of the GUI system is based on current web browser technology (e.g., Netscape or Microsoft Explorer) to run on low-cost computer platforms.

The Data Warehouse Subsystem is notified by the System Monitor when new data are available. Before the data files are loaded into the DW, a Preloader process will dramatically reduce the all points telemetry data into change-only and statistical averages. Additionally, a set of ancillary data, e.g., spacecraft and science instruments events and orbital parameters, are incorporated into the DW archive. The main purpose of the Preloader is to convert the 3 terabytes per year of all points telemetry into a more manageable data volume of about 100 gigabytes per year that can be more easily ingested into the DW. The DW is designed for complex query capabilities to detect dependencies across spacecraft subsystems, to detect trends, and to classify performance of HST subsystems.

Physical Architecture

The CCS Telemetry Processing and Archiving system can be broken down into three separate subsystems. (See Fig. 2: CCS Telemetry Archive Architecture):

- Archive Subsystem;
- Data Warehouse Subsystem; and

- Data Server Subsystem.

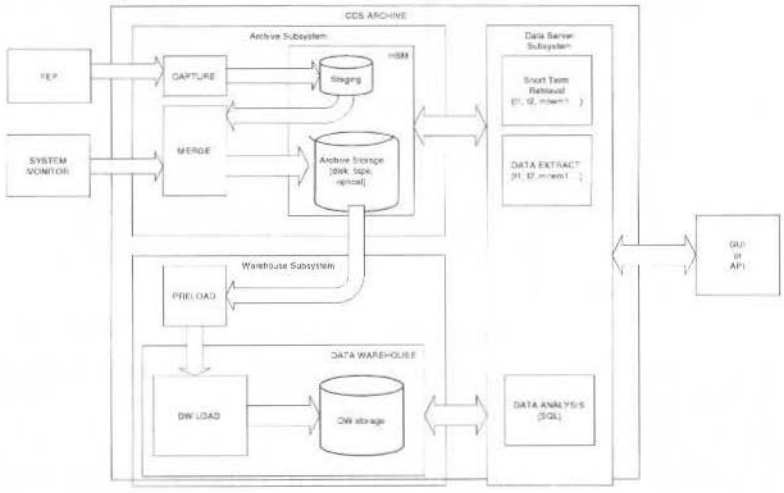


Fig. 2 Telemetry Archive Architecture

Engineering telemetry is managed as a pipeline process beginning as a data stream from the FEP. Within the FEP, HST telemetry data are converted into Engineering Units, and time-stamped with the true receipt time. This process includes automatic detection of changes in the telemetry formats. Converted engineering data are routed to a staging disk on which the files are intermittently stored until the System Monitor system determines the data ready for the merge process. The merge process combines real-time telemetry with data from tape-recorder dumps. The resulting contiguous merged data stream is stored in the all points archive, realized as a disk storage system and backed-up by an automated tape system. This telemetry repository is designed for data analysis over a short period of time with limited query capabilities. (The design goal is approximately 30 days before back-ups to tape.)

The Data Server Subsystem allows the end-users to interface with the All Points Archive, the Data Warehouse, and the database which keeps track of the status and location of the different data products. The Data Server submits the user requests to the archive or the DW and formats the received data for transmission to the user GUI. The Data Server may constrain or even deny requests depending on the user profile, spacecraft and or ground system status.

STOCC Transition

STOCC facilities and equipment were originally designed with some consideration to human factors. The many technical and management changes to the operations environment typically superceded human factors. By the time of the VISION 2000 effort, HST personnel recognized that the configuration of the control room and the layout of the equipment can play a major role in the success of failure of the mission execution.

The HST operational area can be a very stressful environment during anomalies, servicing mission preparation, and servicing missions. During the latter, the engineering support personnel and managers process and analyze complex engineering data; important decisions must be made in real-time and without error.

The STOCC refurbishment gave the opportunity to improve conditions in the HST mission operations environment and thus reduce some of the stress levels.

Process to Improve Human Factors

A systematic process was developed to identify and address areas of concern regarding the configuration and layout of the operations environment.

Surveys

The actual end-users of the facilities (Flight Operations Teams, and system engineers and managers) were surveyed to identify problems with the facilities and the working space environment. The issues

identified included the workspace, the existing console layouts, the noise levels, and the line of sight for personal communications.

Survey Evaluation and Analysis

The survey responses were weighted according to the amount of time individuals had spent in the facilities planned for improvements. For instance the FOTS virtually live in the Mission Operations Room (MOR), so their concerns about MOR were given more importance.

Requirements Development

Based on the weighted results of the survey, a set of functional requirements was developed to specifically address the issues raised by the end-users.

Preliminary Layouts and Designs

A set of preliminary facility layouts and designs were prepared for evaluation by the corresponding experts.

Consultation

There were several facility and console changes that required consultations with facility engineers and human factor experts. accommodating the basic requirements.

Development of Designs

Once the consultants and experts optimized the designs, several review meetings were held with the end-users to validate whether the design satisfied the actual needs.

Implementation

This was the most critical part of the process, given that changes had to be made without impacting the on-going operations and other engineering activities.

Feedback

After the final implementation, the end-users had the opportunity to fine-tune the changes.

STOCC Refurbishment

The STOCC refurbishment took about 6 months. Each operations room required the following changes:

- Removing existing equipment;
- Demolishing the ceiling;
- Removing carpet floor tiles;
- Re-cabling power distribution units to accommodate the new CCS;
- Installing new vinyl floor tiles;
- Installing soundproof ceiling tiles;
- Installing sound proof panels on the walls;
- Installing new consoles;
- Installing new ground system equipment;
- Installing new software;
- Installing multimedia equipment and displays; and
- Testing and Certification of the facility.

The refurbishment required operational support functions to be consolidated temporarily in a room set up and certified as an operational area. The Mission Support Room (MSR) became the pivot facility to during the STOCC refurbishment. The System Engineering and Evaluation Room (SEER) was intensively used by the ground system testing engineers for CCS validation and verification. Once the SEER was completed, the Flight Operations Team moved into SEER while the MOR/SMOR was being refurbished. . Careful planning minimized the impact on on-going operations activities and testing of the new ground system.

Console Design

Because the control room console is the workspace where the FOTs spend most of their time, its design and configuration are significant. The survey identified several critical human factors related to console design. The following are some of the human factors considered in the design of the consoles:

Workspace

In the normal operational environment, there is a significant amount of documentation in the form of log books, timelines, and operational procedures. The work surface needed to accommodate the documentation. "Elbow room" was also required to give the individual operators a sense of personalization of the workspace.

The number of consoles that were required to support servicing mission activities was also relevant to the console design. More consoles are needed during servicing missions than during normal operations.

Several designs were developed to meet flight operations needs and servicing mission requirements. The final design was a compromise between the flight operations needs and the number of consoles that the facility could accommodate to support the servicing missions.

Adjustable Workspace

The consoles were designed with adjustable workspace surfaces. This gave users the ability to adjust the console as needed. This also addressed the need to comply with the American Disability Act (ADA) for accessibility of facilities and working areas.

Lighting

The HST flight operations personnel prefer to monitor telemetry and operate at an ambient lighting of approximately 30 lumens. However, this level of intensity is inadequate for writing and reading of operational documentation. This conflict was solved by providing each console with a dimmable light source directed toward the work surface and, to avoid glare, away from the video display unit.

Glare

Glare from extraneous scattered light can cause significant visual interference to console operators attempting to read small fonts on CRTs. Glare can also cause eye fatigue, which compounds the difficulties caused by the glare itself. The new consoles used special designed glass that avoids or minimizes direct glare, reflection from the CRTs, and the console lights.

Seating

Seating posture and comfort affect the level of physical fatigue that a console engineer may experience during long hours of operational activities. Thus chairs specified for the final configuration had several adjustable factors to accommodate individual preferences.

Facility Layouts

Early in the design of the facilities it was determined that it was necessary to optimize the facilities to meet the number of consoles and support personnel.

Different operations required different interactions among operations, and access to different documentation or equipment. The nature of these interactions, especially direct communications, played a significant role in the layout of the HST facilities.

Acknowledgments

Many thanks to Joe Pollizzi, Space Telescope Science Institute, who never tired of answering questions concerning the technology and the concept of the Data Warehouse. Special thanks also to William Sawchuck, Space Telescope Science Institute, who was generous with his time as a technical writer when reviewing and correcting this paper.

References

Gainsborough, J., August, 1997, "Control Center System Overview". NASA/GSFC.

Rifkin, A., 1997, "Special Report: Reengineering the Hubble Space Telescope Control Center System". Institute of Electrical and Electronics Engineers, Inc.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 275

ISSN 0100-7386
Printed in Brazil

Data Processing Systems II

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp.276-285

ISSN 0100-7386
Printed in Brazil

STDS - Satellite Telemetry Dissemination Services

S. Alonso
P. Honold
J.M. Martinez
J. Noguero
GMV S.A
Isaac Newton 11, 28760
Tres Cantos, Madrid, Spain
sara.alonso.renedo@esrin.esa.it
phonold@eutelsat.fr
jmmf@gmv.es
jnoguero@gmv.es

P. Guerrieri
J. Maurissen
D. Paris,
M. Rulent
EUTELSAT
Rue Balard, 70
Paris, France
pguerrieri@eutelsat.fr
jmaurissen@eutelsat.fr
dparis@eutelsat.fr
rulent@eutelsat.fr

Abstract

In 1998, the EUTELSAT Spacecraft Control Centre (SCC) includes a new component providing satellite data availability at the desktop, the Satellite Telemetry Dissemination Services (STDS). The STDS is a whole new approach, providing a flexible and comfortable World Wide Secure Access to both Real-time and Historical Satellite Data. The STDS specifies a thin layer for distributed TM access, and implements a modern Client to monitor and process Telemetry (TM) at the desktop under Windows NT, taking full advantage of this environment.
Keywords: *Spacecraft Control System, Satellite Telemetry Monitoring, Software Maintenance, Layering, Client-Server Architecture, TCP, Desktop, Security, Windows NT.*

Introduction

In the early days of the EUTELSAT II program, access to satellite telemetry was only possible by physically entering the Satellite Control Centre (SCC), and using a dedicated workstation. This system was rather inflexible in terms of ease of access to satellite data. A facility called the Satellite Telemetry Monitoring System (STMS) was developed in 1992 to allow users to access telemetry from a single workstation. The STMS re-uses the SCC software but is configured to run in a single workstation with Telecommand (TC) facilities disabled. Being the STMS a monitoring component (passive SCC component), it is not constrained to be located within the SCC. The STMS requires X25 access to the Groundstations.

In the frame of the EUTELSAT programme for new developments and SCC maintenance, GMV, S.A. has developed the STDS. The STDS provides satellite data availability at the desktop under Windows NT on the office LAN. This means that end-users (spacecraft engineers, satellite manufacturers, management, etc) can work in their own office PCs (using email, word processing software, Spreadsheets, etc) and access real-time and historical telemetry from the same environment.

Fig. 1 shows the relationship between the different SCC components:

- Mission Server.- This SCC component is an AlphaServer with OpenVMS running the Telemetry (TM) and Telecommanding (TC) software components, the user interface runs in dedicated workstations within the same OpenVMS cluster. A Mission Server allows to control all satellites within a specific satellites family. There is a Cluster of a Mission Server and one or more Workstations for each satellites family. Current satellite families are: EUTELSAT II, Hot Bird, W24, and SESAT (SESAT will become operational in 1999).
- STMS.- It is an AlphaStation running the TM software component together with its User Interface (UI). No TC is available.

- STDS.- The STDS includes two components, a Server component that runs on a Mission Server or on a STMS, and Clients that run on PCs under Windows NT.

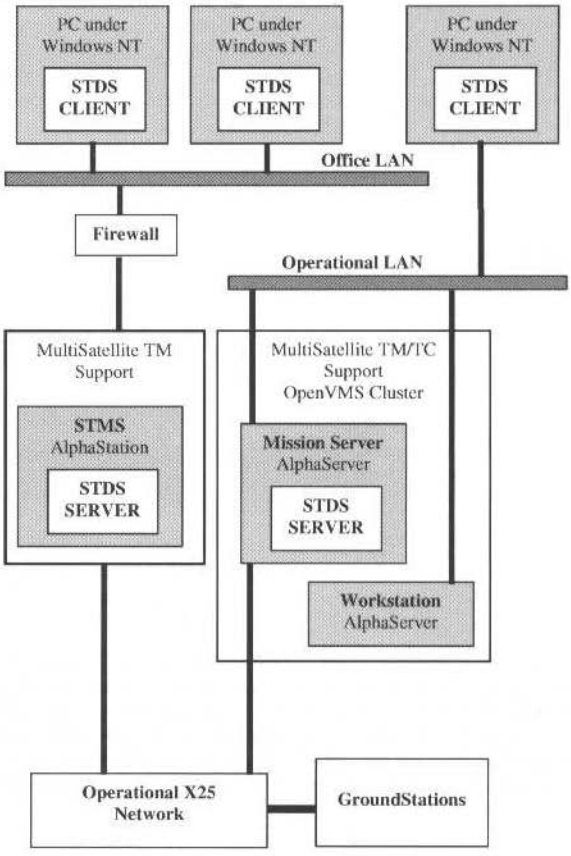


Fig. 1 The STDS vs. the SCC

STDS High Level Components

The STDS specification leads to a design including the following high level components:

- A Telemetry Server that processes TM acquisition requests from STDS Clients. A single Server can serve telemetry for all satellites within a specific satellites family.
- An Administrator tool running on the server side allowing to enable/disable/monitor STDS Clients access and configure key server parameters.
- A formalised TCP/IP-based application protocol for any STDS Client to access satellite data. This interface is formalised in an Interface Control Document.
- The STDS Client for TM storage, analysis, and monitoring. A single Client can access telemetry for all satellites and all satellite families.

The Interface Control Document (ICD) is the essence of the STDS. Any server complying with this specification is able to serve STDS Clients. In the same way, any Client complying with the ICD will be able to access telemetry (once access is granted on the Server). The STDS Client is just a specific type of Client that is part of the STDS specification.

The STDS Server

The Telemetry Server is designed to smoothly integrate with the target system. The server can run on any STMS or on any SCC Mission Server. The server has the following software interfaces to the SCC:

- Events Logger.- To log Server events.
- Telemetry History Files.- To access historical TM (also referred to as "Retrieved Telemetry").
- Inter-Process Communication services.- for accessing the Real Time Data Stream.
- Configuration Tables.- To obtain system configuration information.
- Distributed Database.- To extract and interpret Telemetry Parameters.

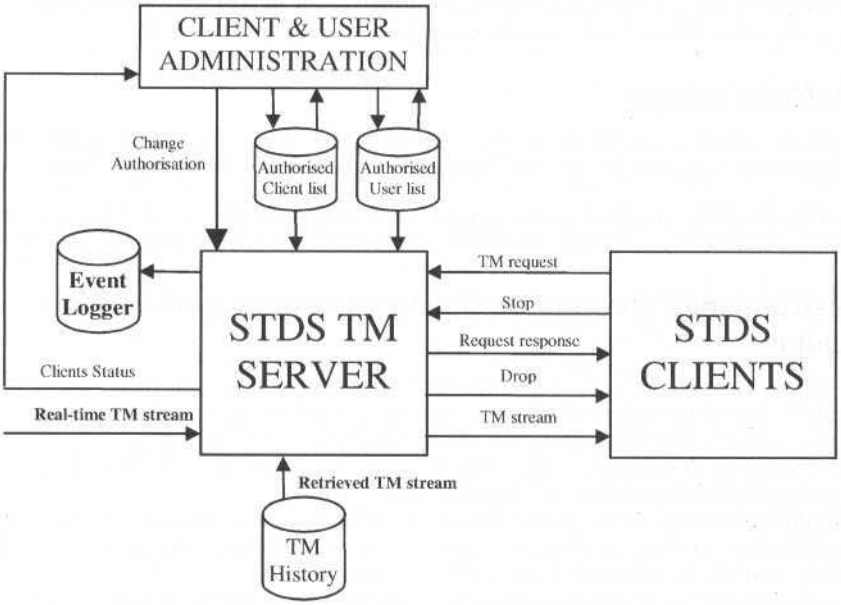


Fig. 2 STDS Server overview

Figure 2 shows the STDS Telemetry Server including its interfaces to the SCC, to the STDS Server Administrator, and to the STDS Clients.

The STDS Client

The type of STDS Client included in the STDS specification incorporates the following functionalities:

- Definition, activation, monitoring, and control of Telemetry Acquisition proformae.
- Definition, activation, monitoring, and control of Telemetry Display proformae
- Definition, activation, monitoring, and control of Out Of Limits Values proformae.
- Monitoring and retrieval of system events through an Events Logger.
- Local storage and playback of acquired Telemetry.

Client Telemetry processing begins after successful activation of an acquisition proforma for a specific satellite; in this activation, the client sends a request with the list of TM parameters specified in the proforma to the server. Once the request is successfully validated, the Client begins to receive messages containing telemetry parameter values. Received values are stored on disk as they are received (if recording has been specified for the acquisition). Message processing triggers a refresh cycle the active Alphanumeric Displays (ANDs) and Graphical Displays (GRDs) to show the new values for the specified satellite; in addition, it triggers Out Of Limits (OOLs) checks.

Design Issues

Most of the design issues included in this section are referred to the STDS Client. Notice that the server was developed to smoothly integrate with the target design on the Mission Servers/STMSs, and was constrained by the target architecture.

Some of the design elements presented are a direct consequence of specific requirements, others are the result of pure design choices.

Client Development Language/Application Framework

During the early development phase, several options were considered for the implementation language. Java, Tcl/tk (Tool Command Language / ToolKit), and Visual Basic were considered as an alternative, or in combination with Microsoft Visual C. The final choice was Visual C, mainly due to performance considerations. This ensures that the system meets EUTELSAT's requirements and enables its future use in other missions with more demanding Telemetry rates.

Design Methodology

The STDS Client is designed using Object Modelling and Design Technique (OMT). This is due to the fact that Object Oriented Design methodologies are suitable for the selected development language (C++).

Object Oriented Design allows abstraction and encapsulation of global data structures; this permits a flexible design and loosely coupled components that are not tied to specific forms of data representation.

High Performance Through Component Integration in a Single Application

This design decision included trade-offs between a design based on independent processes and extensive IPC (Inter-Process Communication) support, or a highly integrated set of components running under a single process. The latter option was selected for the following reasons:

- Performance.- All objects see the same process address space. Data access and method invocation is very efficient, avoiding in this way IPC overhead.
- Multithreaded processing.- Multithreaded Operating Systems such as Windows NT allow to have independent processing threads within the same operating system process. This enables components to execute concurrently within the application.

Loosely Coupled Acquisition Component Using Threads

Since all client application components run under the same operating system process, it is required to de-couple the TM acquisition from the UI, so UI processing does not block ongoing TM acquisitions. This was achieved by including TM acquisition in an independent process thread.

World Wide Client - Server Access

The original design included the use of User Datagram Protocol (UDP) broadcasting to distribute real-time telemetry. This would lead to great savings in Network bandwidth when many Clients request real-time telemetry for the same satellite.

Since UDP broadcasting relies on a specific network infrastructure that propagates broadcast messages as required, this design decision was withdrawn and replaced by the selection of TCP as unique method for Client-Server communication. TCP allows flexible access from anywhere on the Internet.

Client-Server Communications Optimisation

The Client-Server Communications has been optimised by sending only the values for telemetry parameters that change with respect to the last message sent to the client. That leads to a low network load during operations with respect to the calculated nominal network load for a given number of parameters in an acquisition.

Further optimisation can be achieved via compressed data encoding in the presentation layer.

Multiple Document Interface (MDI)

The Multiple Document Interface (MDI) is widely used in today's most popular Windows applications. The MDI allows a very flexible way for a user to configure the layout of the UI elements. It also allows quick access to all system functionalities. In addition, all the application windows are logically grouped within the application main window.

The Visual C application Framework includes the Microsoft Foundation Classes (MFCs). The MFCs support the development of MDI based applications through its predefined object classes.

Scalability

The STDS is highly configurable. Parameters such as the amount of information that can be accessed by a Client, the amount of information that can be served by a STDS server, or the number of displays that may be activated at the same time can be easily modified (via the Administration tool on the server side or via the Registry under Windows NT). The software is designed to allow increasing these limits taking full advantage of upgrades in the underlying hardware infrastructure (Computers and Network).

The network load can be directly controlled on the server side by dynamically modifying the overall maximum number of parameter values transferred per minute from the server to a connected client.

Orthogonal Design

The original requirements were already pointing in this direction. In fact, the processing associated to each proforma type (Acquisition, OOLs, ANDs, GRDs) includes the same design elements. This leads to an orthogonal design and to great flexibility at the UI level. Users may select TM for 10 parameters, and only apply limits to 2 of them, whilst displaying some other 2 parameters.

Another important design element is the access to locally stored telemetry. The STDS allows to replay this TM just like incoming TM from the Mission Servers. This makes all Client processing consistently available for playback TM, and reuses all design elements that are already present for Real-time and retrieved TM (retrieved TM refers to TM that is present in History Files on the Mission Servers or STMSs).

STDS Client User Interface

The User Interface includes elements found in modern desktop applications such as framed windows, choice of interaction via menu options (drop-down and pop-up), toolbar, or short-cut keys; definition and load of work-spaces, and access to local proformae or common proformae stored in file server or remote hosts.

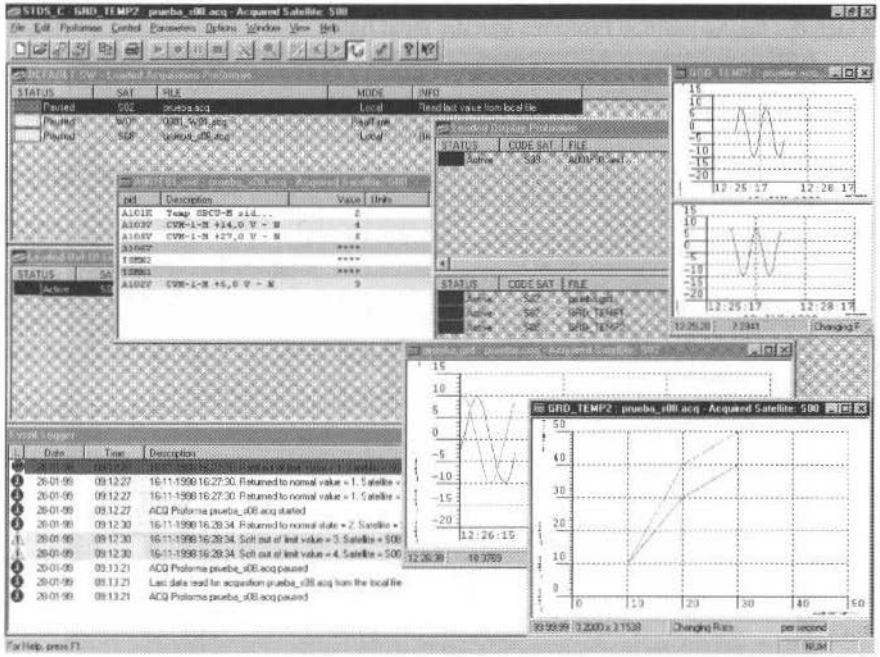


Fig. 3 STDS Client User Interface

Fig. 3 shows the STDS Client Multiple document interface. All application windows are windows frames within the MDI.

The User Interface provides an orthogonal solution for handling proformae. The system supports the following types of proforma:

- Acquisition Proformae.- List Telemetry Parameters to be acquired.
- Alphanumeric Displays (ANDs) Proformae.- List Telemetry Parameters to be displayed.
- Graphical Displays (GRDs) Proformae.-List Telemetry Parameters to be graphically displayed.
- Out Of Limits (OOLs) Proformae.- List Limit checks to be applied to acquired Telemetry parameters.

Acquisition

Fig. 4 shows the common proformae menu options for proformae creation, modification, loading, and activation.

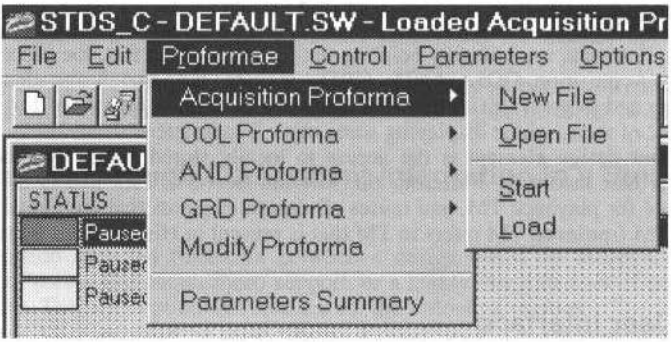


Fig. 4 Consistency in User Interface

In addition there is an extra type of proforma defining lists of parameters that may be directly added to other proformae. Fig. 5 shows an example of proforma definition dialog box (Acquisition Proforma). The proforma includes a set of operations (accessible via buttons) that are common to all proformae. In addition it includes the Acquisition Proforma specific input fields.

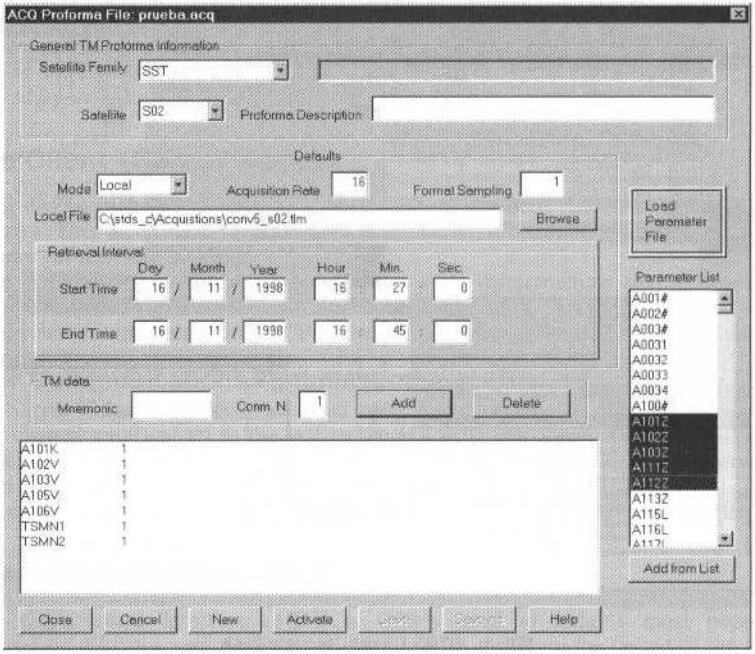


Fig. 5 STDS Acquisition Proforma

In the Acquisition proforma the user may select to acquire real-time telemetry, retrieved telemetry, or telemetry played back from locally stored telemetry. All proformae may be modified and activated

without committing the change, this allows to make small modifications and quickly view the results focussing on the required data. Changes may be saved to disk at any time (Save, and Save As).

Graphical Displays

The Graphical Displays (see Fig. 6) allow to display X/Y graphs for telemetry parameters acquired. GRDs include the following functionalities:

- Display of several parameters versus time or versus another parameter (parameters on Y combined or in a single split frame window).
- Display Freeze to step back and forward to inspect received telemetry values (the number of values held in memory is configurable).
- Export the graph to other Windows applications in Windows Metafile format (vector format).
- Importing of X/Y plot data from other applications to use the STDS as an X/Y graphs visualisation package.
- Zooming, Printing, etc.

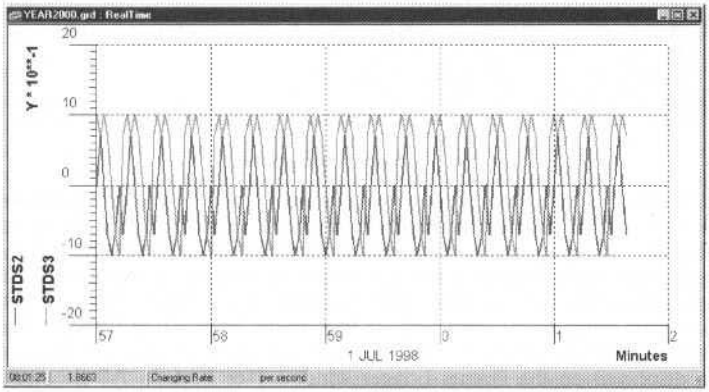


Fig. 6 Graphical Display

Out Of Limits (OOLs)

Limit checks may be applied to incoming telemetry. An out of limits proforma lists the parameters to be limit checked with the corresponding soft and hard limits. OOLs proformae may be activated independently for each satellite.

Limit check violations for acquired parameters raise the corresponding alarms and messages logged in the STDS Client events logger. Parameter values displayed in Alphanumeric Displays use a different background for parameters that are out of limits.

The Events Logger

All STDS Client components log relevant events to the Events Logger. The events are displayed as they are raised in a dedicated frame window (see Fig. 7). Some events include audible alarms for the user to acknowledge.

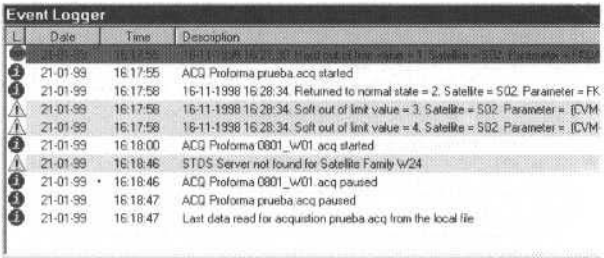


Fig. 7 Events Logger Window

The events logger viewer allows to retrieve previously generated events with a specified filtering criteria (see Fig. 8).

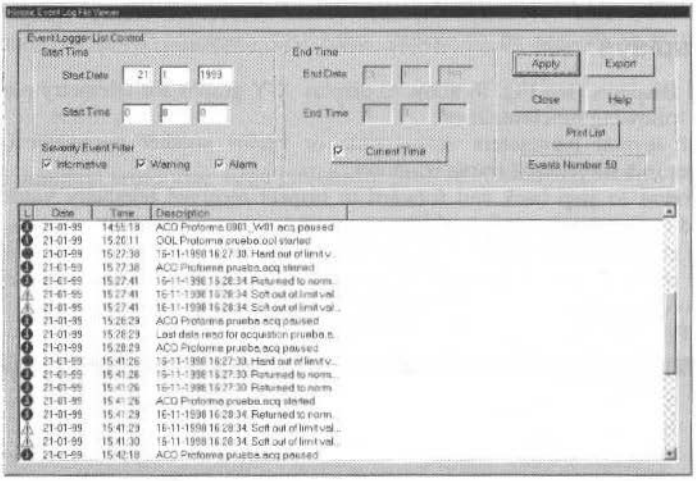


Fig. 8 Events Log Viewer

On top of the STDS client parameters that may be configured by the system administrator, there are many parameters that can be directly configured by the end-user. Fig. 9 shows one of the Client configuration dialog boxes to give an idea of the degree of flexibility available for client configuration.

The "Minimum Disk Space" allows the user to determine the generation of alarms when the remaining disk space becomes less than the specified limit. The "Events Logger Maximum File Size" determines the maximum number of events that are kept in the Events Log Circular File (once the file is full, the latest event overwrites the oldest). The "Maximum number of Events in Logger" determines the number of events displayed on the Events Logger window (see Fig. 7). All the modified values take immediate effect when changes are accepted in the dialog box.

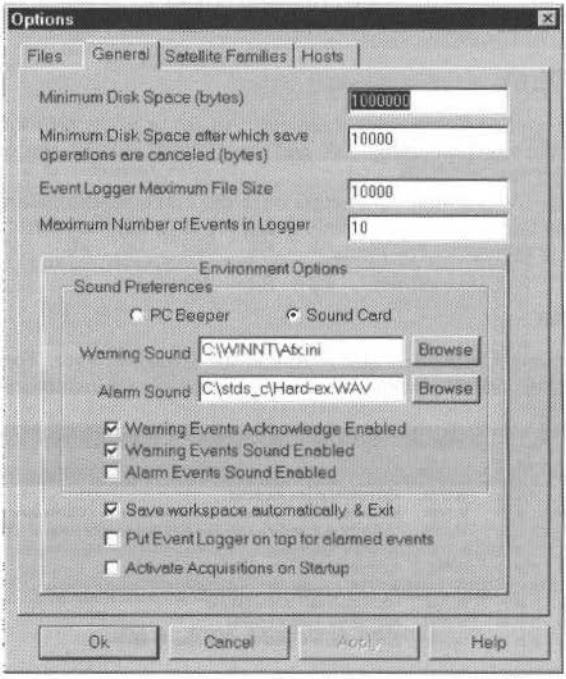


Fig. 9 An Options Dialog Box

The User Interface includes many other features. On-line help is available across the UI with links between help topics.

The Operational Environment Meets the Desktop Environment

A very interesting aspect of the STDS is the confluence of two environments that have traditionally been specified to be de-coupled. This brings up two very different types of issues: Security, and inter-operability with desktop application software.

The STDS includes standard elements to ensure security such as encryption, authentication via signature, and user/host based access control; in addition, the STDS takes advantage of a secure SCC infrastructure layout including X25 Close User Groups (CUGs) definition, firewall protected access, and lack of access to systems running commanding capable components.

The STDS is highly integrated with the desktop environment and allows to exchange information with applications such as Word or Excel; Windows Metafile format is supported for graphics. In addition, it allows importing external plot data in ASCII.

Component Based Approach to Upgrading Satellite Control Centers

The STDS can also be viewed as a real example of component based approach to incorporating modern technology into operational Spacecraft Control Systems. This alternative is very cost effective, and it is based on the identification and grouping of existing functionalities in the SCC and a formalised definition of services required from the SCC in order to implement the required functionalities (layering). Notice that the existing SCC may include software layers that are suitable for upgrading either the layer implementation (maintaining the software interface), or the components that make use of the layer.

This technique has successfully been used in the past in the context of Spacecraft Control Systems. In particular, the European Space Agency Spacecraft Control Operations System User Interface (SCOS-I UI) was successfully upgraded from a screen based UI (using BARCO Workstations) to a distributed X11-Windows based User Interface (running under Sparc Workstations). This distributed UI system is called SCOS-B, and it is currently used for operations.

A more recent example is the upgrading of the EUTELSAT Orbital Operations User Interface from an Ingres forms character oriented UI to Tcl/tk. This led to a radical reduction of system maintenance costs, and the delivery of a very flexible X11-based multi-windows UI.

This approach (layering) is worth considering when making strategical choices about upgrading any SCC infrastructure. Other alternatives, such as implementing of a whole new Spacecraft Control System, are often bound to high cost, and high risk, taking into account the high degree of uncertainty with respect to the stability of available software technologies.

Desktop Telemetry Monitoring System (DTMS)

The STDS concept and design allow its further development and maintenance as a generic Desktop Telemetry Monitoring System (DTMS). This new system will include extra support for all types of missions, and extended capabilities.

Conclusions

The STDS is an off-the-shelf available component that can be incorporated into existing Spacecraft Control Systems. The STDS includes many interesting elements that are worth considering in the context of the development and maintenance of modern Spacecraft Control Systems.

The Satellite Telemetry Dissemination Services were originally conceived to bring the Satellite Data to the end-users desktop. The project successfully achieved this goal through a formalised Client-Server interface specification, and the implementation of a modern client Windows NT application that fully integrates with commercial applications running at the desktop.

The STDS Client has been designed as a lightweight component that is loosely coupled to the SCC. This allows easy installation in a few seconds and minimum requirements on the target platform. No Commercial Off-The-Shelf (COTS) software is required, the STDS only requires Windows NT 4.0.

The monitoring and control facilities implemented on the server side allow wide installation of the Client software on as many target PCs as required whilst keeping control of client access and selecting the desired bandwidth for Client-Server traffic.

The STDS brings more productivity by bridging the operational and desktop environments. These environments have traditionally been specified to be physically isolated. The STDS successfully tackles the new security issues raised.

The project has also been presented as a real example of a component based approach to successfully upgrade a Spacecraft Control System.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp.286-293
ISSN 0100-7386
Printed in Brazil

A Ground System Open Architecture

Han Hwangbo
Joon Kee Min
Tae Ho Park
Korea Telecom Satellite Business Center
206 Chongja-dong Pundang-gu Songnam City Kyonggi-do 463-711, Korea
hhwangbo@kt.co.kr
mjk@kt.co.kr
thpark@kt.co.kr

Abstract

This paper proposes the design of an Open Ground System Architecture with the use of cross-platform middleware and database driven design for building additional database, software and hardware that are distributed across a network for real time satellite command and control. All the real time software processes are divided mainly in two parts; the independent processes of the spacecraft and the other dependent processes of spacecraft. Multiple of these dependent processes are required to support multiple spacecraft. The proposed designs include the real time software and baseband equipment that will accommodate a range of command and telemetry rates and formats. A database management system for straight-forward database manipulation and binary file generator modifications is required. Real time software requires modification to use the command and telemetry database by processing the original database and converting it to more efficient binary files. The proposed architecture provides a functional capability that can support the operation of additional spacecraft from different manufacturer by updating databases and software.

Keywords: *Open Architecture, Middleware, Database Driven, Dependent Processes of Spacecraft*

Introduction

The ground system has to support the following functions: spacecraft commanding, telemetry processing (real-time and development), ranging, orbit determination, communication monitoring, ground equipment monitoring and control. The communication monitoring functions are spacecraft independent and will not be discussed here. The same applies to the ground equipment monitoring and control functions these functions are dependent on the equipment and are not tied to any particular satellite manufacturer. Spacecraft commanding is the most challenging aspect of building a universal ground system that can control satellites from different vendors. Each satellite vendor has different command formats and even satellites from the same manufacturer may have different formats. The telemetry formats for geosynchronous satellite exhibit a high level of commonality. The information is organized by frames and some of the data is sub-commutated. It is our experience that we were able to process a different manufacturer's satellite telemetry without code changes by constructing a proper database. The proposed open system approach provides end-to-end database driven telemetry reception, processing and display.

This paper proposes the design of a Ground System Open Architecture, to provide a functional capability that can support the operation of additional spacecraft from different manufacturer without system redesign. Design has established two distinct and separate environment, the satellite controller environment, and the satellite control software development environment. Design isolates the satellite specific portions of the satellite control system from the generic portion of the system. Design implements the satellite specific subsystem software on its individual hardware. Making the satellite specific software modular. Clearly defining the interfaces between the satellite specific modules, and the rest of the system. The proposed designs include the baseband equipment that will accommodate a range of command and telemetry rates and formats. A database management system for straight-forward command & telemetry database management and modifications is required. An user interface that can be reconfigured by the user without support from the satellite manufacturer is essential. All the Real Time System (RTS) processes are divided mainly in two parts; the independent processes of the spacecraft and the other dependent processes of spacecraft. Multiple of these dependent processes are required to support multiple spacecraft. Real time software requires modification to use the command and telemetry database by processing the original database and converting it to more efficient binary files. The proposed Ground System Open Architecture can accommodate satellites of different manufacturer without system redesign.

Open Architecture Design

The purpose of the Ground System is to provide the ground operations to control the spacecraft during on-orbit operations. To support these operations the Ground System typically includes the Real

Time System (RTS), Operator User Interface (OUI), Command Telemetry System (CTS), Communication Monitoring System (CMS), Ground Status & Control (GSC), Spacecraft Simulator (SS), Automation System (AS), Orbit Determination System (ODS), Network Monitoring System (NMS) and Development System (DS) as listed in "Figure 1 : Ground system configuration modules".

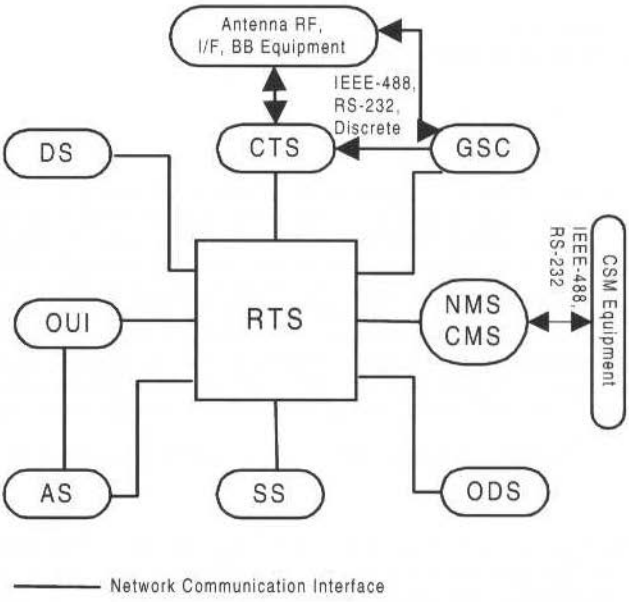


Fig. 1 Ground system configuration modules

Within the Ground System the "Development System" refers to the maintenance and development of satellite control software, done by software developers, on development computers. When software is ready it is installed on an operational satellite control consoles. The arrows show the two configuration modules connected by this interface. The RTS is composed of servers and client workstations for real-time operations. The servers maintain the databases and are focal points for all logging and archiving functions. They also act as communication hubs receiving telemetry from the CTS modules and ground systems and re-distributing telemetry to all clients. It is the RTS server computer that connects to the CTS, OUI and NMS etc.

Middleware Architecture

The proposed open architecture design emphasizes the use of interface standards[1] and the use of implementation that conform to those standard interfaces to support the communication among the various application programs. The proposed interface is a message oriented middleware architecture allowing programs to interact in a synchronous or asynchronous mode through the use of message. The middleware manages all communication among client processes in distributed applications. The client processes can be located anywhere throughout the network and communicate transparently. The middleware uses the publish / subscribe concept for information distribution. Client processes "publish" their information to the server and other clients may "subscribe" to receive the information. The receiving client may start or stop receiving information at any time.

"Figure 2 Cross platform middleware architecture" shows the client/server relation for the OUI, the CTS and the new platform. The OUI has a client that sends the OUI requests to a real-time server and has a client that receives the satellite data and RTS status. The application code on the OUI, RTS computers uses the middleware supplied API. All the processes that have embedded middleware code and interface directly into middleware are referred to as a RTS client. RTS clients communicate with each other through another middleware application called RTS server as shown in Figure 2. Note that each RTS has its own RTS server for redundancy.

Real Time System (RTS) Software

"Figure 3 RTS software design" shows the overall RTS software design. All the boxes represents data structures while all the ovals represents processes or software configuration item running on separate computer. All the network communication interfaces are highlighted. All other interfaces are either memory access, disk access or some other way of inter-process communications such as mailboxes and queues.

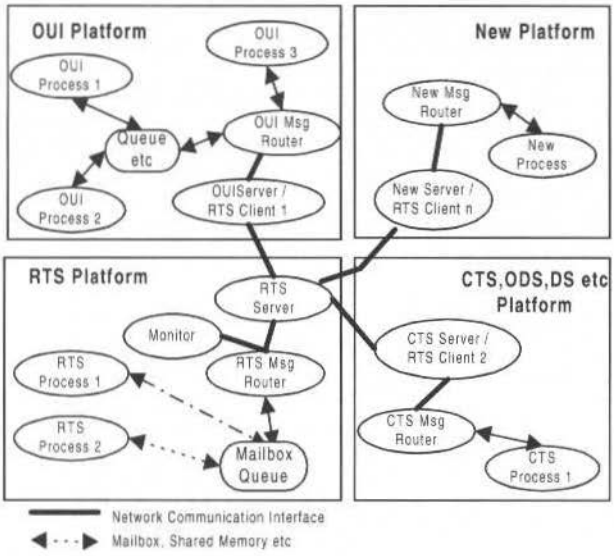


Fig. 2 Cross platform middleware architecture

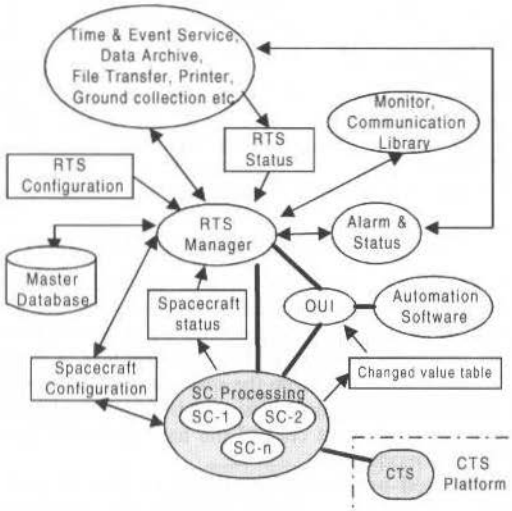


Fig. 3 RTS software design

All the RTS processes[2] are divided mainly in two parts.

General processes that are independent of the space-craft. These are mostly utilities that support the space-craft dependent processes. And space-craft processes these are dependent on the spacecraft being processed. Multiple instances of space-craft dependent processes are running to support multiple spacecraft.

"Figure 4 Spacecraft process software design" shows different spacecraft processes and their interfaces. The processes for one space-craft does not interact with processes for the second spacecraft. It is therefore possible to change any or all of these processes without having no effect on processing of rest of the spacecraft. The critical operations of spacecraft commanding and telemetry processing are for most part database driven. Since RTS and OUI[3] is independent computer software configuration item it is possible to change one without effecting the other as long as they both comply with the interface specifications.

When sending commands, the Command Build Gateway process converts each command in the command list packet to an entry (of command type) in a time ordered linked list. A command database and spacecraft parameter file is required for each spacecraft. This file contains command mnemonics, opcodes, dataword specification rules, command types, and hazardous command information. A set of command parameter files exist in each spacecraft process module. These files are used during the initialization and during the execution of commands. All the parameters discussed in the hardware section are controlled through these files. The files are automatically generated from a master database that resides on the off-line support workstations and is down-loaded to the RTS module during installation or as necessary. The design approach was to build a universal front-end box that can be easily re-configured for different commanding schemes. The commanding and telemetry board was designed with an open system approach, all functions/parameters are controlled via the firmware residing on the spacecraft processes database.

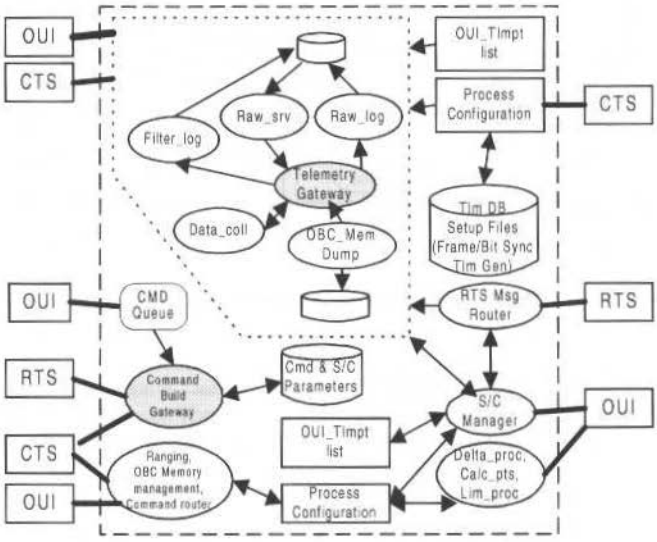


Fig. 4 Spacecraft processes software design

The commands to a CTS module are transmitted over a standard network interface and in addition to the command list they include the spacecraft identification. During command processing, the commanding software dynamically selects library functions based on the name of the spacecraft. Library functions include personality functions that are needed for plain text or encrypted command processing. The library functions are compiled separately from the rest of the RTS software and are loaded during the startup. Changes in the library function do not require re-build of the basic RTS module software. The library functions and the parameter files that control the hardware settings represent the databases and software that personalizes the RTS module.

Once the telemetry is processed by the telemetry system, the internal telemetry representation is independent of the source of telemetry. Also, spacecraft telemetry processing functions such as logging, archiving, archive retrieval, trending, playback and hard copy generation are identical to ground station telemetry functions.

Because of the way hardware platform links separate compilation units together during system startup, the spacecraft specific functions do not have to be linked with the other elements of the RTS software during the software build process. Thus it is easy to add additional spacecraft types without affecting the rest of the software. Developer or third-party supplied spacecraft specific functions can be added by merely editing the RTS load script to include the new functions among the exiting ones. Only a

few RTS header files need to be used by the developers of the new functions. These header files contain command packets, command library, spacecraft database library, telemetry packets and maybe a few others.

Command Telemetry System (CTS) Software

When receiving commands, the Command Executive task monitors the time stamp of the commands in the command list, and dispatches the command when the command time is reached as depicted in "Figure 5 CTS commanding software dataflow". If the spacecraft being commanded has an encryption device, the command is sent to the Encryption task, where the clear text is converted to the cipher text. The command is then sent to both the Send Command task, which outputs the command to the command generator board, and the CV task, which places the command in a pending CV list. The CV task receives telemetry, and uses whatever CV information that is present in the telemetry to verify the command has been received or rejected by the spacecraft. The CV task outputs an event message regarding the status of the command, and notifies the command executive task of the status of the command. This either allows the command executive task to issue the next command, or may cause it to abort or suspend commanding. The Encryption Task also may process telemetry to determine if an encrypted command has been accepted by the spacecraft. Encryption authentication is sent to the CV task.

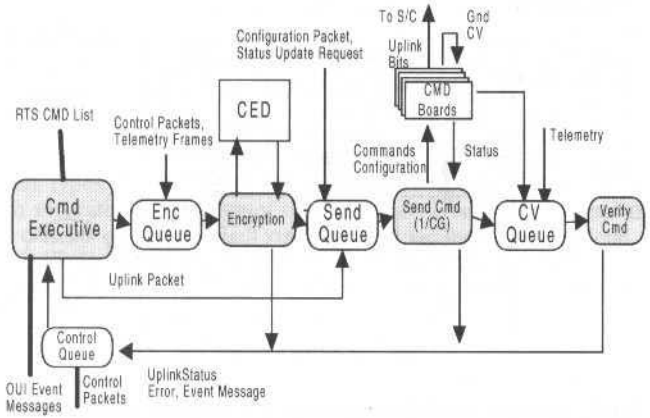


Fig 5 CTS commanding software dataflow

When the command is dispatched by the command executive, the binary form of the command is computed by a function unique to the type of spacecraft. Command Executive function creates the binary format of the command to be uplinked, including sync pattern, decoder address, opcode, datawords, parity, etc. The Send Command Task will send the bits generated by this functions without regard to whatever fields the command may or may not contain.

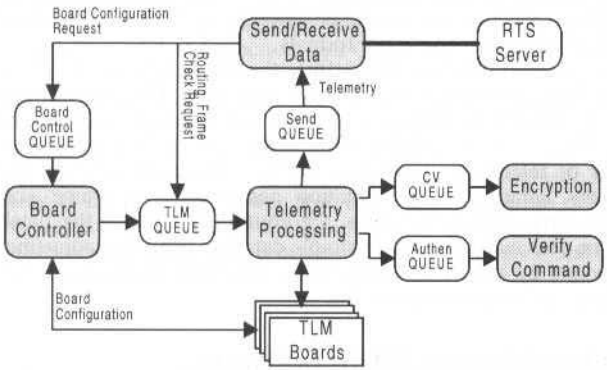


Fig. 6 CTS telemetry software dataflow

"Figure 6 CTS telemetry software dataflow" shows different telemetry processes and their interfaces. The CTS telemetry software design approach is introduced by the usage of a database manager. Telemetry formats and processing information will be stored in a relational database RTS module. The telemetry hardware provides high level of integration and flexibility. The telemetry boards were integrated with an open system approach, all functions and parameters are controllable from the RTS module software. The CTS module telemetry collection is fully controlled through databases. Parameter such as frame size, synch pattern, telemetry rates are programmable and a new telemetry format requires only minor changes in databases.

Command, Telemetry and Spacecraft DBMS

RTS/OUI/CTS and ODS software uses the command and telemetry database by processing the original database and converting it to more efficient binary files. This method has two major advantages. First, since binary file generation can be done off-line, the data is prepared ready to be used by RTS/OUI etc in more space-efficient on-line processing manner. Secondly, if there are changes in the original database, it is possible to adapt or reformat the database such that the final binary files contains little or no changes in the format. This can eliminate or minimize changes required on RTS etc software. "Figure 7 : Database management to support multiple types of spacecrafts" shows database management processes and their connectivities.

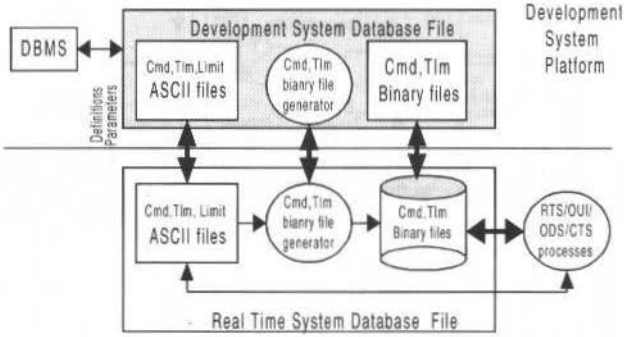


Fig. 7 Database management to support multiple types of spacecraft

The user input validation and command list generation are driven by databases. Each spacecraft has its own database that defines the characteristics of the command structure, the content of the command and the effect of the command as reflected in telemetry. The command effects are used for functional command verification. Each command database is controlled through the database manager. The proposed design provides a tool to retrieve the information from the development database and to create binary command databases used by the real-time system. The process of creation of the binary databases and distribution are invisible to the user. All the aspects of command structure such as; op-codes and data-word mnemonics, individual bit field definitions, command length, number of data-words spacecraft addresses and preamble bit patterns are controlled through the database manager thus the commanding process can handle a wide variety of command formats.

Each spacecraft has its own database that defines the characteristics of the telemetry structure. The databases are controlled through the off-line database manager. The proposed design provides a tool to retrieve the information from the development database and to create binary telemetry databases used by the real-time system. The process of creation of the binary databases and distribution are invisible to the user. All the aspects of telemetry processing, such as mnemonics, individual bit field definitions, calibration curves, state names, sub-commutation and sub-frame, frame size, information stream identifications and telemetry sync bit patterns, are controlled through the database manager thus the telemetry process can be personalized through simple database telemetry adjustments. The user addresses individual telemetry points by mnemonics; any other additional information can be retrieved using the standard DBMS delivered by the database vendor.

Orbit Determination System (ODS) Software

The Orbit Determination System (ODS) software was designed to be as independent of spacecraft design and therefore capable of supporting any three axis geosynchronous satellite regardless of

manufacturer. Differences in individual satellites are handled through parameter databases as described below "Figure 8 : ODS software design".

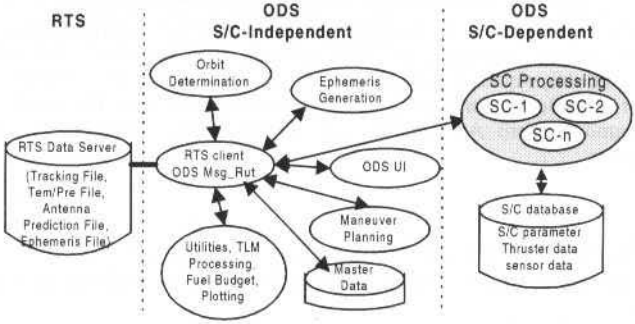


Fig. 8 ODS software design

The solar pressure force is a function of the spacecraft area to mass ratio as well as the reflectivity properties of the various spacecraft surfaces. The ODS provides a database driven multiple surface model which allows the spacecraft to be modeled as a collection of independently defined surfaces for the purpose of computing the solar pressure force. The sensor events are a function of the sensor type and position on the spacecraft. ODS supports a variety of database driven sensor models typical to three axis geosynchronous spacecraft. These models can be easily modified by database parameter adjustment or extended with additional plug-in software components where required. Satellites from different manufacturers and even different generations of satellites from the same manufacturer have a distinctive complement or thrusters types, thruster locations, and thruster pointings. The proposed design provides an open system which separates the physics of the maneuver from the individual thruster performance. Many thruster types can be accommodated via database parameter setup alone, however additional models can be easily plugged-in if necessary. Once the propulsion model is setup for a particular spacecraft, it will fit seamlessly into the rest of the ODS maneuver planning software.

Evaluation

To evaluate scope of changes to support different manufacturer's spacecraft, we consider to make necessary changes to the RTS/OUI/ODS etc software utility to adapt the new format to existing format. A good evaluation here may eliminate some of the changes required by subsequent steps, and since the database binary flat file utility is standalone it is relatively straight forward and risk free change that can be made to the system. The command database binary file generator required change to reflect change the basic command data structure. OUI command editor required changes on OUI to accommodate different data structures on OUI. There is however no change required on look and feel of the OUI screen. Telemetry database has relative small changes and required changes in the following areas. The telemetry database binary file generator required change to accommodate difference in the input data file. The telemetry processor spacecraft process will need minor change in telemetry processor. The data collection processes will require change. The onboard computer memory management software (OBC table uplinks) will require change as the flight software is completely different between the two types space-crafts. OUI software needs changes in the area of data collection, and OBC table uplink. All spacecraft configuration displays will have to be created for the new spacecraft. All the changes required are in the area of the spacecraft processes and can be done without effecting the operation of the other spacecraft processing. There is positively no change required in basic software design or as far RTS / OUI is concerned no additional hardware is required.

Based on the changes required in the RTS/OUI databases the second step is to see which of the ODS, AS etc processes are likely to change. The answer here may vary from all to only a few with minimal changes. The most likely candidate for change is ODS processes and minor change to the AS process. The biggest issues are ODS sensor difference, propulsion system difference and updating model from telemetry data.

Conclusion

This paper has described a design for using Ground System to guide the process of implementing Open Architecture. The proposed design can be used to control additional different types of spacecrafts. Based on the database driven commanding and telemetry processing, many of changes can be handled by

changing the databases and middleware interface utility for inter process communications. The spacecraft dependent files are automatically generated from a master database that resides on the off-line support workstations and is down-loaded to the operational modules during installation or as necessary. All spacecraft dependent functions are supported by individual asynchronous processes. It is convenient to adapt these processes without effecting the other spacecraft. Packet based interface between RTS and other configuration modules makes it possible to change RTS processes without changing the recipient process on the other module, as long as the output is compliant with the interface specification. The proposed Open Ground System Architecture can accommodate satellites of different manufacturer without system redesign.

References

[1]Talarian Corporation, Mission-Critical Interprocess Communication, 1997.

[2]Martin Marietta Astro Space, KOREASAT Real Time Software Design Document (DD-RTS-20032250), 1995.

[3]Martin Marietta Astro Space, KOREASAT Real Time Software Workstation Display Software CSCI Test Plan (PN-RTW-20032250), 1993.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp.294-302

ISSN 0100-7386
Printed in Brazil

Samos - A Flight Dynamics Full Mission Support System for Artemis

M. A. Molina
J. Potti
G. García-Julián
J. R. Píriz
GMV S.A.
c/ Isaac Newton 11, 28760
Tres Cantos, Madrid, Spain
mmolina@gmv.es
jpotti@gmv.es
ggarcia@gmv.es
jpiriz@gmv.es

G. di Genova
Telespazio
Via Corcolle 19, E-00131
Rome, Italy
fds@mephisto.sdc.asi.it

Abstract

*The Satellite Attitude Monitoring and Orbit control System (**SAMOS**) was developed for the ARTEMIS program as an operational Flight Dynamics System (FDS) supporting the full satellite mission from launcher injection until satellite end of life. This system will cover all the activities related to orbit determination and control, attitude monitoring, on-board control subsystem support and payload support during both LEOP (Launch and Early Orbit Phase) and GEO (geostationary) phases. A software vendor independent rationale has been selected. An overview of the architecture of this system follows, along with the advantages of the Full Mission support concept.*
Keywords: *Flight Dynamics System, Full Mission.*

Introduction

ARTEMIS (Advanced Relay Technology Mission), currently scheduled to be launched in the year 2000 on a Japanese H2A rocket, is an advanced geostationary ESA satellite designed for testing and operating new telecommunications techniques and services. ARTEMIS will carry two **payloads** for direct communication between satellites. These payloads will receive data from low- Earth-orbiting (LEO) satellites and transmit them directly to Europe. They are a laser-optical relay terminal called Satellite Inter-Orbit Link Experiment (SILEX) and a double-frequency S-band (2 GHz)/Ka-band (23 to 27 GHz) terminal called S-band/K-band Data Relay (SKDR). Optical frequencies have the potential for very high capacity and for secure communications using small terminals. ARTEMIS will also carry an advanced L-Band land mobile payload providing two-way communications via satellite, between fixed Earth stations and land mobiles (trucks, trains or cars) anywhere in Europe and North Africa. An EGNOS navigation payload is also included as part of the ARTEMIS services. This payload will allow delivery of EGNOS messages in order to ensure the required accuracy and integrity requirements at user level.

ARTEMIS will fly advanced platform technologies in addition to the communications payloads. These technologies include the **ion propulsion** package which will apply ion propulsion for 10 years providing North/South station-keeping to demonstrate its operational capability for the future missions. **Users** of ARTEMIS will include, among others, the French Earth Observation satellite SPOT 4, NASDA's telecommunications satellite OICETS, ESA's ENVISAT and possibly the International Space Station.

The Satellite Attitude Monitoring and Orbit control System (SAMOS, see Fig. 1) was developed by GMV for Telespazio, as an operational **Flight Dynamics System** supporting the full satellite mission, from launcher injection until satellite end of life. This system will cover all the activities related to orbit determination and control, attitude monitoring, on-board control subsystem support and payload support during both LEOP and GEO phases.

SAMOS provides the optimum orbit transfer manoeuvres required during LEOP and the orbit control and wheel desaturation manoeuvres required during GEO, using both chemical and ionic propulsion packages. SAMOS is also able to estimate on-ground the satellite attitude during all the mission phases by interpretation of the available on-board sensors measurements (e.g. Infrared Earth Sensor, Coarse Sun Sensor, Precise Sun Sensor and Gyroscopes). Additionally, SAMOS integrates some specific on-ground

algorithms used to generate the parameters required to configure correctly different on-board subsystems, in particular, the satellite attitude control subsystem.

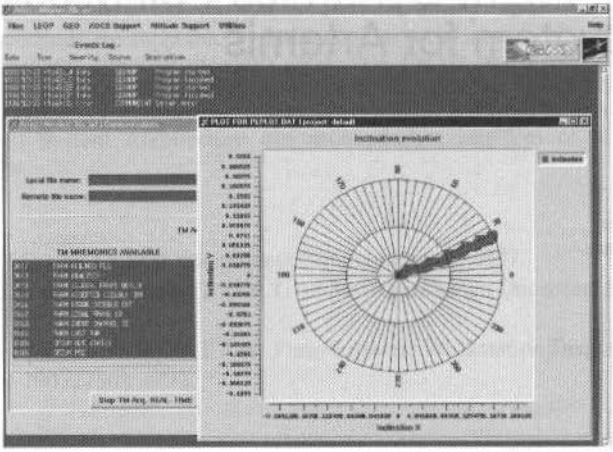


Fig. 1 SAMOS

Payload activities should be highlighted since they are especially relevant for ARTEMIS. Among other things, SAMOS is responsible for the validation of the final payload operations schedule, verifying its feasibility regarding all the mission constraints.

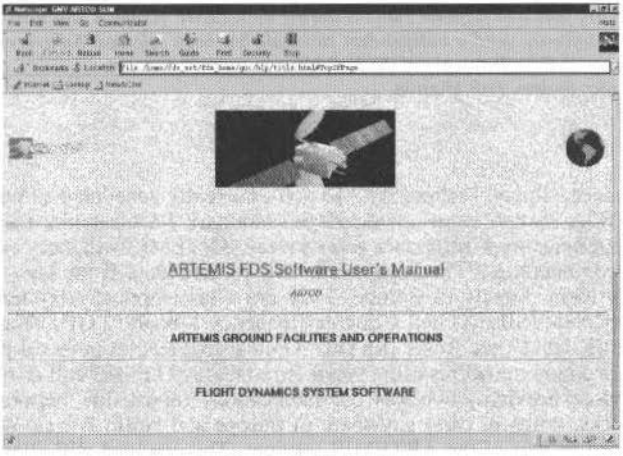


Fig. 2 On-line help

Communications between SAMOS and the Satellite Control Centre (SCC) follow a client-server architecture supported by the RPC protocol. SAMOS is able to receive from the SCC in near real-time the requested telemetry (TM) parameters and tracking measurements needed by the attitude monitoring and orbit determination processes respectively. On the other hand, SAMOS computes all the required information to monitor current satellite status and provides the SCC with all the required parameters (via data files) as input to the telecommand generation subsystem.

SAMOS is developed following **a vendor independent rationale**. The operational software runs on a PC-based platform under LINUX, equivalent to the development environment. FORTRAN 77/ANSI C are the main programming languages for algorithms implementation and communications purposes respectively, using the *g77/gcc* compilers and *gdb* debugger GNU development tools. The user interaction is based on a user friendly state-of-the-art Graphical User Interface developed using Tcl/Tk. The on-line help (see Fig. 2) is a set of HTML pages that can be viewed with any standard browser. Direct links from each application window to the corresponding HTML pages are established.

An overview of SAMOS follows, highlighting the advantages of the FDS **full mission** support concept.

Samos Architecure Overview

General

SAMOS is designed as a **highly modular** system in order to allow regular updates of the system, adaptations to other missions and/or migrations to different platforms.

The **Graphical User Interface** (written in Tcl/Tk) can execute any of the independent computation programs (written in FORTRAN 77) and controls the Communications Module (written in C) as well as some general utilities.

Three different **environments** are provided: Operational, Simulation / Training and Maintenance. Simulation modules are not available in the operational environment. Different projects can be created in each workstation / environment combination. Some data synchronisation functions across workstations, environments and projects are provided.

The **main functions** that are provided for both LEOP and GEO operations are shown in Table 1. In many cases they are implemented through different programs for each phase in order to take into account the characteristics of each orbit type (LEOP/ GEO).

Table 1 SAMOS Main Functions

Function	LEOP	GEO	Comments
Raw Tracking Simulation	X	X	For simulation and training only. Includes measurement errors. Supports ranging, angles, meteorological and Doppler information.
Raw Tracking Pre-processing	X	X	Processes ranging, angles, meteorological and Doppler information, performing reduction, smoothing and calibration of data.
Pre-processed Tracking Simulation	X	X	For simulation and training only. Includes measurement errors. Supports ranging, angles, meteorological and Doppler information.
Orbit Determination and Prediction	X	X	Creates internal orbit file, to be used by many other modules. A complex physical model is taken into account. Biases, manoeuvre delta-vs, solar radiation pressure coefficient and others are estimated with great accuracy.
Orbit Archiving and Retrieval	X	X	Orbital information archiving, compression and search.
Propellant Mass Evolution	X	X	Performs book-keeping of remaining propellant mass and compares with information given by Telemetry.
Products Generation	X	X	Antenna Pointing Elements, STDM (Spacecraft Trajectory Data Message), etc.
Events Detection	X	X	Eclipses, colinearity masks, perigee / apogee passes, etc.
Apogee Engine Firing Optimisation (AEFOS)	X		Computes optimal manoeuvres from transfer orbit to near-synchronous orbit.
Manoeuvre Calibration		X	For both impulsive (chemical thrusters) and continuous (ion propulsion) manoeuvres
Station Keeping Manoeuvre Planning		X	For both East / West and North / South manoeuvres. Both chemical thrust and ionic propulsion supported. This deviates from conventional station keeping
Payload Support		X	ARTEMIS exchange orbit file generation, inter-orbit link simulation (including payload schedule verification), payload calibration support, etc.
Telemetry (TM) monitoring	X	X	Both real-time and play-back telemetry monitoring, in particular for attitude monitoring purposes. Parameters derived from TM can also be monitored.
Attitude Modes Support	X	X	Including SAM (Solar Acquisition Mode), EAM (Earth Acquisition Mode) and NM (Normal Mode)
Gyro Calibration	X	X	For both LEOP and GEO, filters are used to estimate the gyro drift from TM.
Ion Propulsion Support		X	Computation of the adjustments on the ion propulsion platform orientation required to achieve a target torque, using TM info.
Momentum Wheel Support		X	Momentum Wheel Desaturation planning and delta-v estimation.
Additional Functions	X	X	Solar Aspect Angle Monitoring and On-board Orbit Propagation Parameters Computation.

Fig. 3 shows the **High Level Architecture** from a functional point of view. The main computation functions, inputs and outputs are displayed.

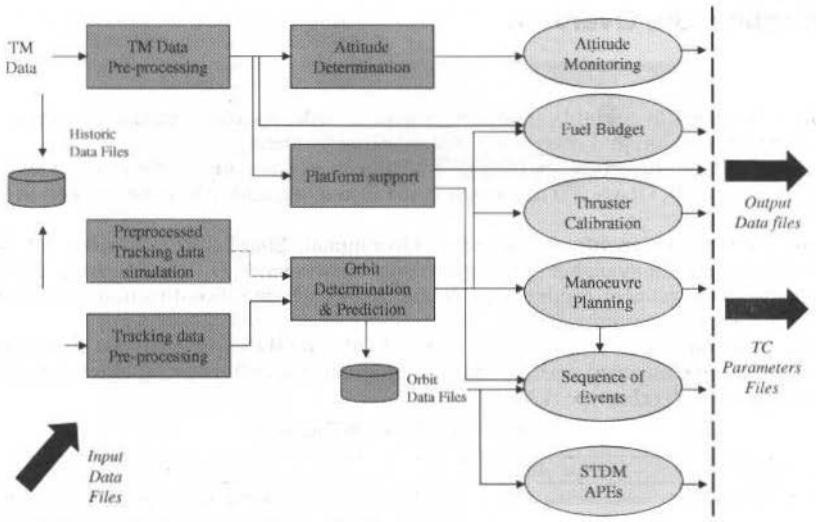


Fig. 3 High Level Architecture

Some aspects of the functions that are common to LEOP and GEO can be highlighted, in particular the **Orbit Determination and Prediction** functions. An iterative batch least-squares estimation process is used in order to determine the state vector at a reference epoch from the pre-processed observations. Station biases, solar radiation pressure coefficient and manoeuvres can also be estimated in both GEO and LEOP. The drag coefficient can be estimated in LEOP. A GEM-T1 (36, 36) gravity model is used for LEOP, while a simplified model (8, 8) is used during GEO.

Fig. 4 shows the architecture for the Payload Support functions. This is a mission specific set of modules of special importance since they compute the ARTEMIS orbit files (A-ORB) that are distributed to all the users in order to allow them to estimate the ARTEMIS position and velocity at a given time. They also check the intended payload operations schedule to validate all the inter-orbit link activities.

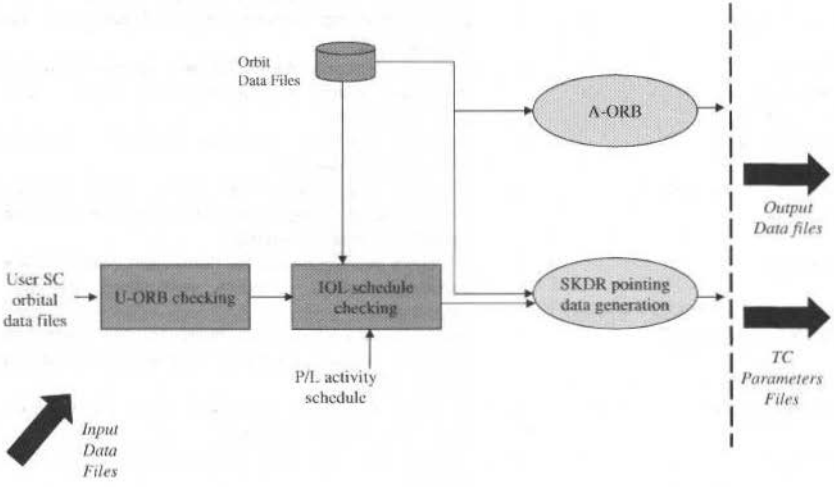


Fig. 4 Payload Support Architecture

From a **user's point of view**, it is easy to access any of the more than 60 computation programs from the main console, through cascade menus. SAMOS programs run independently and communicate through data files. These intermediate files and the input and configuration files used are ASCII in most cases. They can be viewed or edited via user friendly panels such as the one shown in Fig. 5. An in-line help and a help button linked to the corresponding on-line help pages are provided.

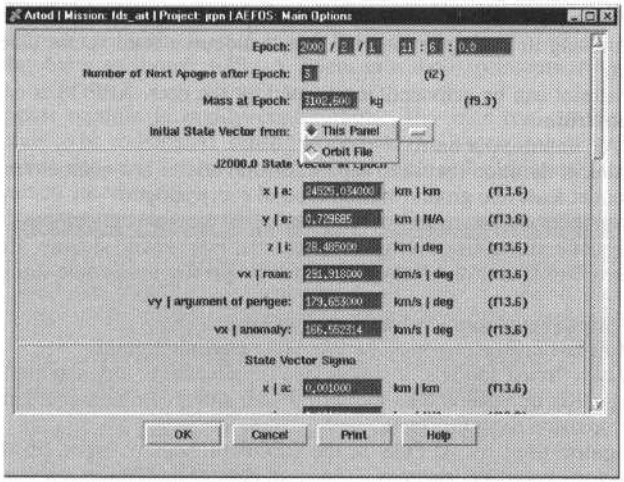


Fig. 5 File edit panel example

Time-tagged execution and communication events (e.g. programs success or failure, FDS-SCC data exchange, etc.) are logged in the main **Events Logger**, which is shown in the main console and can be searched.

LEOP Support Specific Functions: AEFOS

As shown in Table 1, some LEOP specific functions are provided. In particular, a very complex **Apogee Engine Firing Optimisation Module (AEFOS)** is available. It is in charge of the critical mission of calculating the optimal (in the sense of minimum propellant consumption) manoeuvres that take the satellite from GTO (Geostationary Transfer Orbit) to NSO (Near Synchronous Orbit) while (optionally) satisfying the mission constraints. The core problem can be mathematically posed as a Constrained Non-linear Local Optimisation Problem (Non-linear Programming) and is solved by a standard dedicated subroutine.

The optimisation variables (vector **x**) are the parameters that define the apogee manoeuvres. The parameterisation of a manoeuvre depends on the thrust steering policy chosen by the user. Two thrust steering policies are supported by AEFOS:

1. **Inertially fixed thrust:** the thrust vector points in a constant inertial J2000.0 direction.
2. **Rotating thrust:** the thrust vector rotates at a constant rate around an inertially fixed axis.

AEFOS starts the calculations using some initial values of the optimisation parameters. These values are called the *initial guess* and are automatically calculated by a dedicated subroutine which splits the total delta-V based on the transfer strategy selected by the user, keplerian movement and impulsive manoeuvres. The initial orbital elements are propagated through the non-impulsive manoeuvres defined by the optimisation parameters. Once the orbit has been propagated the objective function and the equality and inequality constraints are calculated. New estimates of the optimisation parameters are then calculated by the optimisation subroutine. The process is repeated iteratively until all the constraints are satisfied and a minimum objective function is reached.

AEFOS supports two main **optimisation modes**:

1. **Global minimisation mode:** in this mode no actual target is sought and the only constraints are those mission constraints that have been activated by the user (all of them inequality constraints). The objective function to be minimised is the propellant consumed by the apogee manoeuvres plus the propellant spent in the station acquisition phase. The latter is calculated by a dedicated subroutine that uses as input the state vector after the last apogee manoeuvre. This is the normal operating mode.
2. **Target mode:** in this mode the user can enter the state vector (or some of its components) that must be reached at some time after the last manoeuvre. Only the propellant consumed by the apogee manoeuvres is minimised. The constraints are those mission constraints that have been activated by the user (inequality constraints) and the equality constraints corresponding to the active target components. This mode can be used in case of an emergency and also for manoeuvre calibration.

It must be noted that AEFOS can handle any number of **manoeuvres**. Therefore it can be used to optimise the whole sequence of manoeuvres from separation to NSO and also to successively re-optimise

the remaining manoeuvres after the first ones have taken place. It can also be used to calibrate a single manoeuvre by re-optimising it, setting as target a post-manoeuvre state vector as estimated by the orbit determination program

Each mission constraint can be activated/deactivated by the user. AEFOS is capable of handling the following **types of constraints**:

1. Maximum LAE impulse per burn
2. Maximum transfer duration (between separation and end of last manoeuvre)
3. Coverage from at least one ground station around every apogee
4. Coverage from at least two ground stations during all apogee manoeuvres
5. At least *n* days of continuous coverage from at least two ground stations after the last manoeuvre
6. Angle between Sun vector and body-frame Y axis greater than some minimum value during all manoeuvres
7. At least two orbital revolutions between manoeuvres
8. Longitude after last manoeuvre must be West of a target longitude

AEFOS supports two **thrust models**: a simple one (applicable to any satellite) in which the LAE is modelled just by its specific impulse and its mass flow rate and a complex one that fully implements the ARTEMIS Unified Propulsion System.

The **orbit propagator** used by AEFOS is the LEOP propagator from SAMOS, which has been seamlessly integrated for overall consistency. The user has the possibility to configure the amount of detail with which the orbit perturbations shall be modelled: These range from a simple keplerian orbit to one of the highest accuracy.

AEFOS is able to carry out **stochastic optimisation**, in such a way that confidence levels can be reported about the minimised fuel consumption and about the achievement of the constraints. The stochastic parameters that AEFOS can take into account are the uncertainties of the initial state vector and the uncertainties in the magnitude and direction of the LAE thrust.

GEO Support Specific Functions

Table 1 shows all the functions provided for the GEO phase in SAMOS.

Both **chemical and ionic propulsion** are supported for Orbit Determination and Prediction, Manoeuvre Calibration and Station Keeping, (Spindler, 1994) purposes, which is not conventional in an FDS.

Mission specific **Payload Support** modules have been developed. They play a critical role in the mission generating crucial information for the Inter Orbit Links (IOLs) between ARTEMIS and LEO satellites, as mentioned earlier in this document. Analytical models, (Morley, 1991), are used to approximate the orbits of the two satellites involved.

AOCS Support

The functions related to AOCS (Attitude and Orbit Control Subsystem) support included in SAMOS are shown in Table 1 .

TM Monitoring

SAMOS also provides TM monitoring functions that can be used for, among other tasks, **attitude support**. Telemetry parameter values can be monitored both in **real-time** and **play-back** (using historic telemetry information).

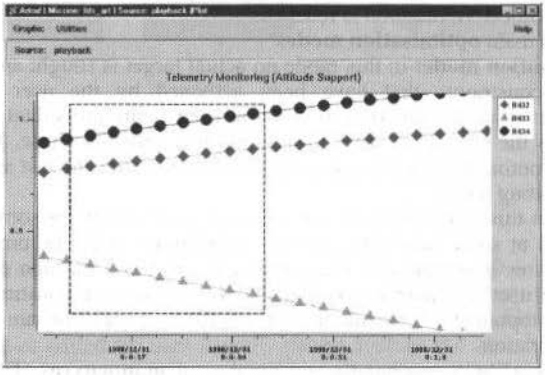


Fig. 6 TM Monitoring

Attitude information can be generated for all the possible attitude modes using the telemetry information for the sensors applicable to each mode. Individual telemetry values can also be monitored in both real-time and play-back, as shown in Fig. 6. Zoom-in and out, printouts and window move functions are available.

The attitude support module is highly configurable. The user can configure the graphics to be displayed, attitude mode to be considered, sensor properties, etc. Numerical values of the parameters (TM and computed variables) can also be monitored in real-time and play-back (see Fig. 7). An Attitude History File is maintained automatically.

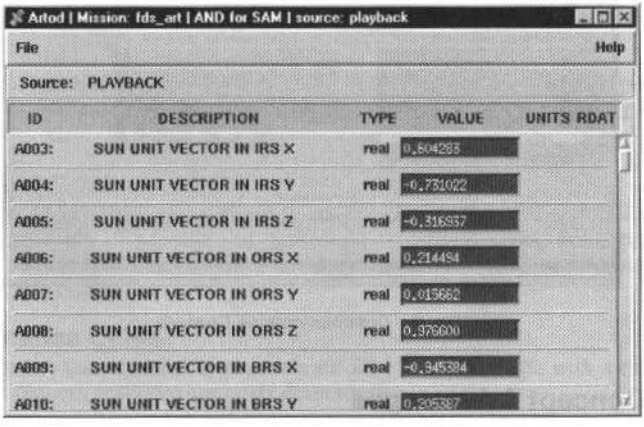


Fig. 7 Telemetry Monitoring

General Utilities

Additional general utilities are provided by SAMOS, as shown in Table 2.

Table 2 SAMOS General Utilities

Project Management	Each project has an independent set of data files to allow experimentation (different cases)
File Manager	General file management functions
Application events	Automatic book-keeping, search function
Data Export / Import	Allows data exchange and synchronisation of workstations, environments and/or projects.
Communications functions	Send/request files to/from SCC; configurable telemetry acquisition
Conversion functions	State vector to/from TLE (Two-Line Elements); state vector and orbital elements transformations across several reference systems
Exchange file generation (to be sent to the SCC)	Events database search and exchange file generation, manoeuvre exchange file generation.

Communications

Communications with the SCC are carried out through the **RPC** (Remote Procedure Call) protocol, following a client / server architecture. Several workstations running SAMOS can send concurrent requests to the SCC.

Fig. 8 shows the **communications console** where the user can activate requests for TM (real-time, playback, individual values for a given time), as well as send and request data files.

Additionally, a **permanent process** running on the workstation receives and stores tracking data sent by the SCC, as well as data files sent at any time. This service is independent from SAMOS in order to ensure that the data sent by the SCC is stored even if the application is not being run.

Telemetry data is stored in a **TM History File**. This file has been designed with a very flexible data structure using linked lists for each parameter which allows it to integrate data received in playback and real-time. Data gaps can easily be filled in this way. Separate files for each hour simplify disk house-keeping.

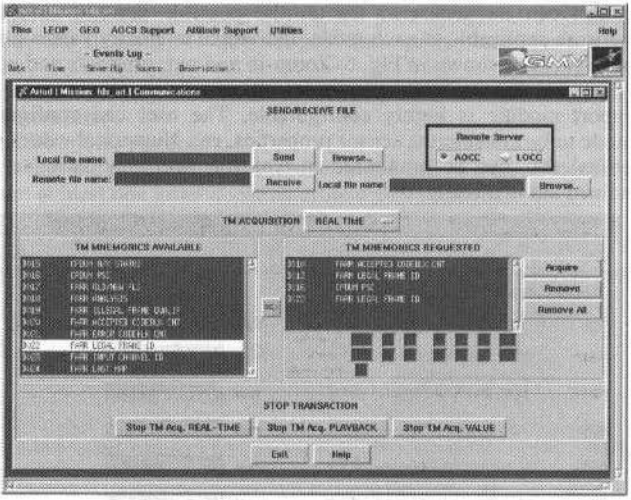


Fig. 8 Communications Console

Full Mission Concept Advantages

It is tempting to try to fulfil all the required functions of a Flight Dynamics System through the use of a set of independent tools, a combination of COTS (Commercial Off-The-Shelf), in-house specific applications customised for the mission, legacy code, spreadsheets, etc. Each of these parts provides the functionality required for a given part of the mission. Usually, different applications are used for the LEOP and GEO phases.

One could argue that this is the most cost-effective approach. The experience of the authors of this paper shows, however, that **a very modular architecture** like the one used for SAMOS **allows an easy integration** of legacy well-known code together with mission specific new code. State-of-the-art User Interface languages such as Tcl/Tk achieve a high productivity not seen before.

A very versatile **toolkit for the rapid development of Graphical User Interfaces** (GUIs) has been created by GMV. It allows a quick integration of new modules in the application, modifications and the adaptation of the system to other missions. An example can be seen in the Payload Support modules in SAMOS, which have been developed ad-hoc for ARTEMIS and have been integrated in the system seamlessly and quickly.

Table 3 Common modules for all mission phases

Communications	A single communications module can be used for all mission phases and types of data (files, tracking, telemetry)
Libraries	General purpose libraries (orbital mechanics, mathematics, etc) provide consistency.
Application Events handling	Consistent application events handling
Graphical User Interface	Same look and feel for all parts of the system.
Telemetry History File	Handling of this file is done by the same module for all mission phases
Mission events database	A single library is used for this purpose for all mission phases.
Common computation programs	Tracking simulation and pre-processing, Attitude and AOCS modules are common for GEO and LEOP (a flag in the input indicates the mission phase in order to take into account some effects).

This **flexible architecture** approach ensures that the software can be adapted to the evolution of the requirements through the whole mission (nominally 10 years) since modules can easily be added, removed or altered without affecting the rest of the system and without having to stop it. The fact that Tcl/Tk is an interpreted language allows changes in parts of the user interface without having to stop the application.

A seamless integrated system ensures consistency in the computations performed. Many modules are shared by all the different parts of the system, as shown in Table 3.

Many Flight Dynamics Systems provide LEOP support through customised software that is independent from the GEO software and not portable to other missions. SAMOS is unusual in the sense that it **integrates LEOP and GEO support** with a modular architecture that makes it easy to be adapted to different GEO satellites.

The system consistency and complete integration has a huge impact on overall operations quality and costs that should be taken into account when planning for a mission. They are described in the following sections.

Improved Data Quality

Common libraries and services improve data consistency and reduce the possibility of errors. Important data files (gravitational field, Earth rotational parameters, etc.) are not duplicated and can easily be updated for the whole system.

Reduced Operating Costs

A single, consistent system with a user-friendly interface and a complete and easily-accessible on-line help allows a reduced number of operators to perform all the Flight Dynamics operations successfully. An operator will not need to be an expert in all fields and will be able to be responsible for more tasks than in a heterogeneous system.

Reduced Maintenance Costs

A single configuration management system is used for the whole FDS, with a rational and seamless architecture. This fact reduces maintenance costs dramatically.

Reduced Training Costs

Understanding of a single and consistent system with common services for all mission phases is a lot easier than understanding the relationship among a bunch of heterogeneous applications. The on-line help of SAMOS works so that the relevant HTML pages can be accessed from each application with a single mouse click. A consistent system reduces the learning curve for the operators.

Conclusions

The Satellite Attitude Monitoring and Orbit control System (SAMOS), developed as an operational Flight Dynamics System in the frame of the ARTEMIS mission has been presented. An overview of the architecture from a functional point of view has been provided.

The advantages of the Full Mission Concept over a heterogeneous system have been presented. A single, consistent system provides improved data quality as well as reduced operating, maintenance and training costs.

State-of-the-art graphical user interface programming languages such as Tcl/Tk allow an easy integration of legacy code with mission specific software. Evolution of the software in order to adapt to the changes in the requirements during the whole mission is also made easier.

Adaptation of this system to other GEO satellites would be extremely easy, as well as extensions for multi-satellite support.

References

Spindler, K., February 1994, "ARTEMIS Longitude Station Keeping". ESA/ESOC OAD Working Paper No. 541.

Spindler, K., August 1994, "ARTEMIS Inclination Station Keeping". ESA/ESOC OAD Working Paper No. 444.

Morley, T. A., July 1991, "A Spot Orbit Model On Board ARTEMIS and SPOT-4". ESA/ESOC OAD Working Paper No. 444.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 303

ISSN 0100-7386
Printed in Brazil

Real Time Systems II

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp.304-309

ISSN 0100-7386
Printed in Brazil

GSOC – Integrating Packetized Telecommanding and Multimission Cross Support

Norbert Jansen
German Aerospace Center (DLR)
German Space Operations Center (GSOC)
Oberpfaffenhofen
D-82234 Wessling
norbert.jansen@dlr.de

Simon Maslin
ANITE Systems
Im Leuschnerpark 4
D-64347 Griesheim
Simon.Maslin@DLR.de

Abstract

This paper describes the approach taken by GSOC to upgrade the capabilities of its telecommand system to allow support of missions that apply the new ESA Packet Telecommand standard. In addition it takes a deeper look at the requirements and restrictions imposed by the necessity to provide cross support to an increasing number of external agencies and defines impacts on the telecommand system architecture by illustrating certain typical mission scenarios.
Keywords: *Packet TC, Cross Support, Ground Station Interfaces.*

Introduction

Many of the new generation of satellites to be flown by DLR / GSOC apply the new packet telemetry and telecommand protocols as defined in the ESA Packet Telemetry and Telecommand Standards. Consequently, GSOC had to upgrade its multi-mission control center facilities in order to include the required new functionality.

Although GSOC does not have its own world-wide ground station network, it has cooperation agreements with other agencies such as ISRO and INPE and these allow it to provide cross support services for spacecraft manufacturers, other space agencies and scientific community. When GSOC operates as a mission control center it also requires cross support from other agencies such as NASA or CNES.

The new Packet Telecommand Standard had a major impact on the GSOC Telecommand System architecture, the 'Frame Operations Procedure' (FOP) of the 'Command Operations Procedure' (COP-1) protocol had to be integrated into the existing system. This was primarily to ensure that ESA standard compliance was retained within the GSOC control center itself and also to provide independence from the many ground stations integrated into the support network.

The prime reason for this approach was that GSOC had to be able to provide mission support using the existing all its ground station network facilities, both those belonging to GSOC itself and those belonging to external agencies.

Requirement Overview

The requirements for the implementation of the new ESA Telecommand standard were based on requirements generated not only at GSOC but also on inputs from major clients requiring these services.

These were as follows:

- Retain existing core command systems functionality.
- Implement new ESA Packet TC Features including
  - Full COP-1 support incorporating monitoring and control functions
  - Packet, Segmentation, Frame, CLTU level implementation
  - Authentication
- Retain compatibility with existing ground station facilities
- Provide Service Access Points for external telecommand users

Table 1 GSOC Missions requiring support from external agencies

Project	Type	External Agency	Data-Interface
EUTELSAT W24	Geostationary	ISRO (BGL)	CLTU
LEOP		NASA-DSN	CLTU
		Eutelsat (Simulator/EGSE)	Frame
ABRIXAS	Scientific	NASA-DSN/Polar	CLTU
		OHB (EGSE)	Packet
CHAMP	Scientific	NASA-DSN/Polar	CLTU
		DLR/DJO (EGSE)	CLTU
GRACE	Scientific	NASA-DSN/Polar	CLTU
		DLR/DSS (EGSE)	CLTU

Table 2 Cross-Support Missions

Project	Type	Station-Interface	Data-Interface		Customer
			Input	Output	
EUTELSAT W24 (Station B/U	Geostationary	DLR-WHM	Frame	CLTU	EUTELSAT
ROCSAT-1	Scientific	DLR-WHM + ISRO (BGL, MAU)	CLTU's	CLTU's	NSPO / Taiwan
KOMPSAT	Scientific	DLR-WHM + INPE (tbc)	CLTU	CLTU	KARI / S. Korea

The missions to be supported by the new system can be divided into two categories. Those where GSOC acts as mission control center and those where GSOC provides cross-support (see tables above).

Telecommand System Architecture

During the design phase of the implementation of the ESA Packet standards (Ref. 1) a number of criteria were taken into account. As the current systems were mission proven over many years, maximum re-use of existing systems was specified and then decisions were made as to how and where the required CCSDS Packet Layer services were to be implemented within the existing modules.

The requirements specified that the Command System had to implement the ESA standard up to and including the coding layer. This was later extended to include certain functions of the physical layer, such as the inclusion of the ESA PLOP-2 acquisition sequence. The latter is significant, for instance, during the LEO phase of a particular mission where ground station support is provided by the NASA network. Full Packet layer implementation thus ensured GSOC's ground station independence. In addition to implementing the full functionality, attention was also given to the flexibility of the system and its ability to support the Packet TC functions not only where GSOC was acting as the control center but also were cross-support functions were being provided.

Taking a look at the layer diagram, the aim was to provide access points for all operating modes at each layer. Obviously this depends fully on the requirements of each mission, but this overall goal was set so that one implementation could be used for all mission scenarios. This meant that access should be provided for TC Packets, Segments, Transfer Frames and CLTUs.

The diagram also indicates the option to allow TC Segments to be fed back into the packetisation layer, obviously a non-nominal case. This feature is provided for the Abrixas mission, which has an interface of TC Packets for test purposes and CLTU's for real-time mission activities. The TC Database is set to output TC Segments, TC Packets with TC Segment header. This feature allows the same TC Database to be used during the test and operations phases.

The following diagram defines the structure of the Core Command System.

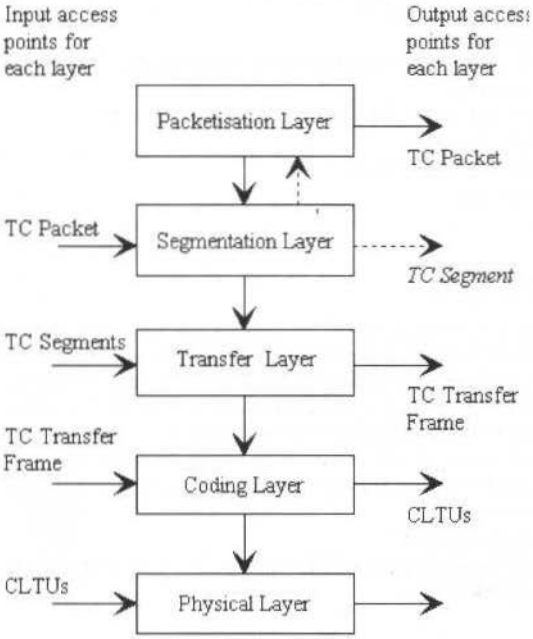


Fig. 1 Processing Layers

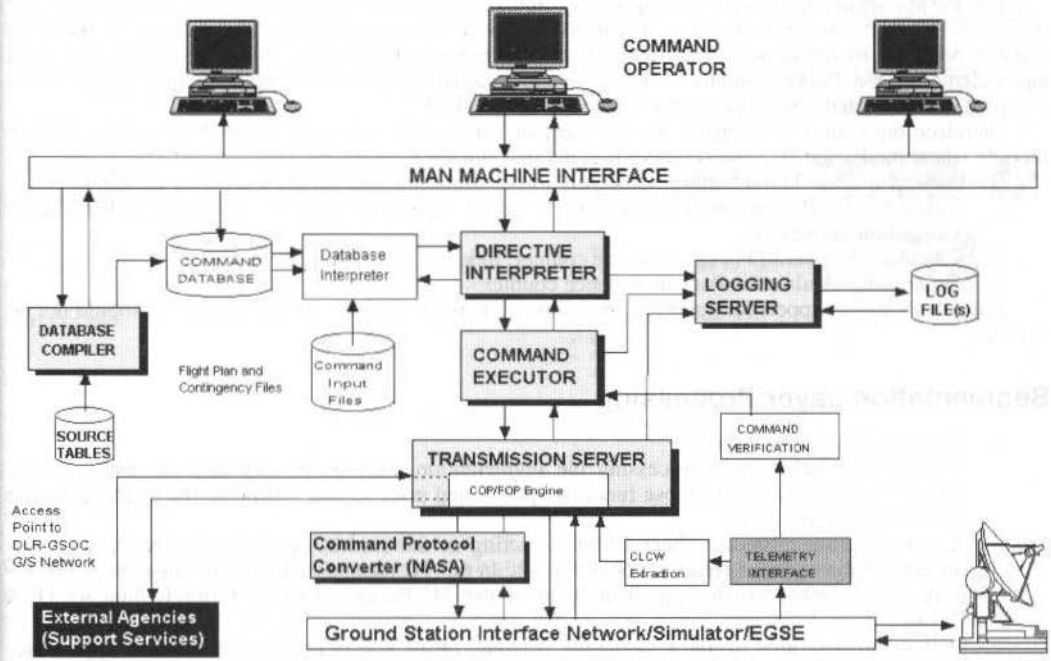


Fig. 2 Telecommand System Architecture

Packet Telecommand Services

The Packet TC enhancements required to be implemented could be broken down as follows:

- Packet Layer Processing

TC Packet Generation, CRC calculation.

- Segment Layer Processing

Segment Header (Trailer) Generation, optional Authentication

- Transfer Layer Processing

Transfer Frame Header Generation, Frame CRC check calculation.

- COP-1 Transfer (sub-)Layer

COP-1 Processing, CLCW Interpretation

- Coding Layer Processing

CLTU Generation

Packet Layer Processing

The functions provided for the packet level processing were divided into two separate activities.

1. Those which were performed in the TC Database following TC selection and prior to uplink.
2. Functions required following enabling of the telecommand for uplink.

For missions where GSOC is acting as the control center the TC Packet is 'fully' defined within the TC Database, with the exception of the packet sequence counter.

- Translate TC mnemonic input into binary data, using database tables. Complete Packet Data field.
- Fill-out Packet Header, Version Number/Type, Data Fields header flag, Application process id,
- Set Packet counter to zero.
- Generate temporary Packet CRC

Once the Telecommand has been successfully translated is then added to the selected sequence/queue, where it is stored prior to uplink.

Packet Sequence Counter generation

The Packet sequence counter is only finally updated once the telecommand has been enabled for uplink. This is performed for a number of reasons. The prime one being that the sequence/queue handling features provided by the Executor task allow translated telecommands in the queue to be moved, copied and deleted. If the Packet counter was set at telecommand generation time then this would allow completely inconsistent sequence counter values to be uplinked.

Therefore once the telecommand has been enabled for uplink it is sent to the Transmission Server module where the Packet sequence counter is generated and the Packet CRC recalculated, if required.

The Packet (re-)calculation feature is also mission configurable and allows 3 separate options:

1. Calculate the Packet sequence counter for each Application Id. As specified in the Packet Utilization Standards.
2. Calculate the overall Packet sequence counter, independent of Application id.
3. Do not (re)calculate the Packet sequence counter

This allows cross-support for agencies providing input as TC Packets where the content should not be altered.

Segmentation Layer Processing

1. Segment Header generation:
   As with the Packet layer processing the segmentation layer processing also has two separate activities which are tied to those functions performed prior to and following the enabling of the Telecommand for uplink.
   Again for those missions where GSOC is acting as the control center is was decided to also implement the segment Header processing within the TC Database. This is because the MAP-ID is nominally related to the Application id of the TC Packet, (CPDU, Prime/Redundant Data Units).
   Options are provided to allow switching between prime and redundant units where applicable. This has been implemented by either duplicating the telecommand within the TC Database or by defining an override switch to allow Map-Id selection in real-time.
   The final part of the Segmentation Layer processing is the optional Authentication and is performed only after the telecommand has been enabled for uplink.
2. Authentication:

Authentication can be switched ON/OFF in real-time by predefined telecommand(s). If it is set to 'ON' the segment trailer (LAC-ID + LAC Counter + Authentication Signature), using Authentication software services of the LETS software library.
If it is set to OFF, no further processing is required.

Transfer and Coding Layer Processing

The processing of these two layers is realised by the Telecommand System using a commercial package (LETS) available in the form of an OpenVMS object library. This package realises the Transfer and Coding layers of the ESA standard and optionally the Authentication layer if required. There are eight different service modules, which allow the user to access and manipulate the FOP. These service modules have been integrated built into the Telecommand System and provide the necessary functionality.

The virtual channels are independently managed in the FOP and have no cross coupling effects. Delivery of invalid CLCW's (i.e. unidentified VC number) will cause all channels to be reset to the initial state).

Due to the fact that this functionality is built into the control center software, no changes to the standard ESA baseband software are required.

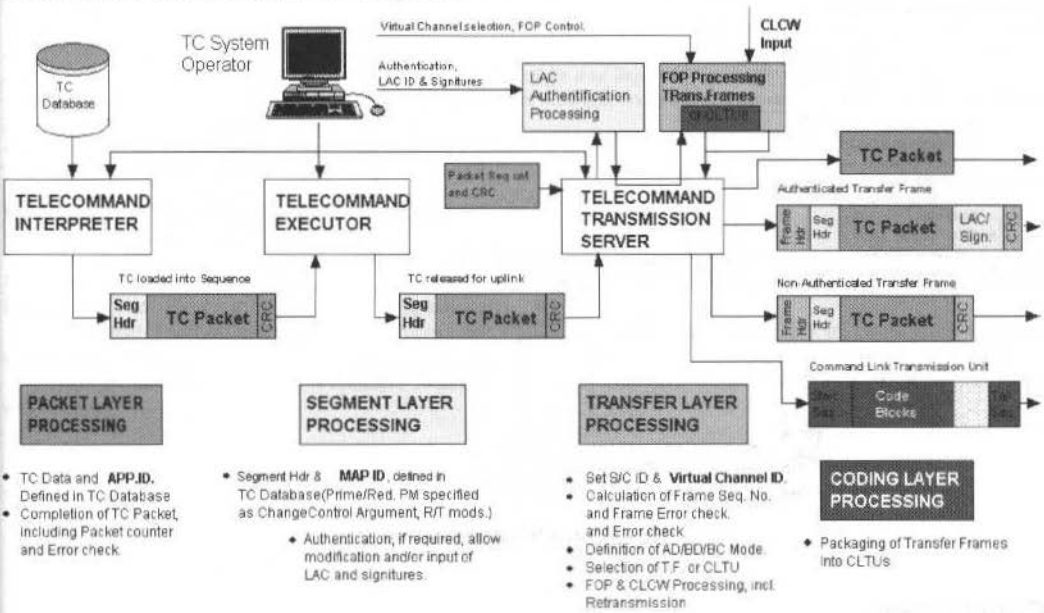


Fig. 3 Packet Telecommand System

Cross Support Aspects

Within the DLR-GSOC ground station network the interface protocols are standardized, enabling the GSOC SCC to communicate transparently with each of its supporting stations. One of the current problems with providing cross support services is the fact that the agencies requiring these services rarely use the same interface protocol. This is a problem that is now being addressed, by the definition and subsequent implementation of Space Link Extension Services, which will provide a standardized interface between the SCC and the Ground Station. For now however, individuality is the norm. During the implementation of the "access points" for each mission it became clear that the best policy was to implement each customers adopted interface definition/protocol locally. In many cases this was a requirement and not an option. However, although the effort for GSOC was increased, this method allowed the customer to treat GSOC in exactly the same way as it would one of its own ground stations or at least allow it to have only the one interface, thereby allowing SCC – G/S independence.

Eutelsat W24:

Support for the Eutelsat W24 project was provided in two forms, firstly during the LEO phase where all activities where performed by the DLR-GSOC control center and secondly in the form of emergency cross-support. During the test and checkout phase of the mission the standard Transfer Frame interface was used. This naturally had one minor drawback that the systems were never completely tested in the final mission configuration, namely with CLTU's.

Rocsat:

Among the recent projects DLR-GSOC has been involved in the support for the ROCSAT-1 mission. This takes the form of cross-support where GSOC provides services to allow NSPO to command through DLR-GSOC's ground station network. Although the satellite uses the CCSDS standards, the interface itself is NSPO specific.

The interface is defined via the coding layer, therefore NSPO provides GSOC with CLTUs or as is the case with this interface a set of CLTUs. GSOC's function is to provide the protocol conversion between the incoming data block format and that required for GSOC network. At the time of implementation buffering features had not been implemented, therefore the throughput was restricted to a "one in, one out" basis.

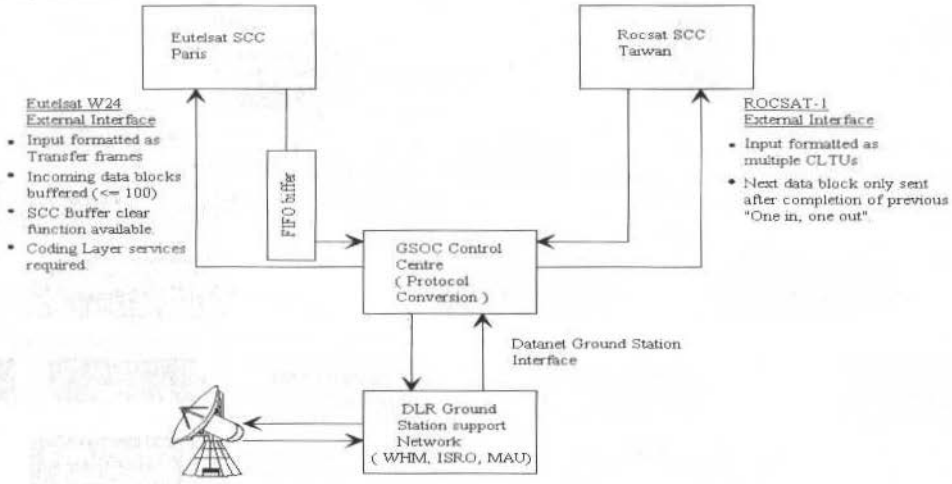


Fig. 4 Cross Support Scenario

Conclusions

With the current implementation of the Packet Telecommand System a great deal of flexibility and independence has been maintained. Flexibility because we have the ability to continue to interface with all existing ground stations in the same manner as before, and independence because as the implementation of all layers has been realized within the control center. This means that no enhancements or modifications at the stations were necessary to support the new standard, although these are obviously been planned and will be implemented in the future. This level of independence means that when the new standards come into force only minor changes will be required within the control center.

As far as cross support is concerned the situation will only improve once a globally accepted standard for data exchange between control centers and ground stations, such as the proposed SLE-Services, has been adopted.

References

[1] ESA Packet Telecommand Standard ESA PSS-04-107
[2] ESA Packet Utilisation Standard ESA PSS-07-101
[3] Telecommand Decoder Specification ESA PSS-04-151

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp.310-317

ISSN 0100-7386
Printed in Brazil

An Object-Oriented Interface to the CCSDS Ground Telecommand Services

Tim Ray
NASA

Jeff Condron
Raytheon
Goddard Space Flight Center, NASA
20771 Greenbelt, MD USA
Timothy.J.Ray.1@gsfc.nasa.gov
Jeffrey.Condron@gsfc.nasa.gov

Abstract

The Telecommand Data Routing and Channel Services defined by the Consultative Committee for Space Data Systems (CCSDS) are flexible enough to support a myriad of commanding models. Because the standard is so broad, the traditional ground system approach has been to implement only the portion of the standard needed by the particular spacecraft being tested/operated.
Tasked with providing ground Telecommand Services for an entire class of spacecraft, where each spacecraft may choose any valid CCSDS commanding model, we designed an interface capable of supporting the full CCSDS protocol. Significant cost savings are achieved by using the same interface and implementation for multiple spacecraft.
This paper describes this interface and how it leads to a straightforward implementation. The concepts used are general enough to be applied to other problems.
Keywords: *CCSDS Telecommand Services, Ground Systems, SFDU.*

CCSDS Telecommand Services

The purpose of the ground Telecommand Services is to reliably deliver your commands to the spacecraft. This is accomplished via a series of layers which add headers and trailers to your command Packet and pass the resulting data to the spacecraft.

The flight Telecommand Service layers then remove the headers and trailers to extract the original command Packet.

Each ground layer accepts a specific form of input data and generates a specific form of output data, as follows:

Table 1 Layers of the Telecommand Services

Layer	Input data	Output data
Segmentation	Packet	Segment
Transfer	Segment	Transfer Frame
Coding	Transfer Frame	Command Link Transmission Unit (CLTU)
Physical	CLTU	command data for the spacecraft

The Telecommand Services are quite flexible. They allow:

1. Data entry at user-specified layers:
   The services allow you to insert data at either the Segmentation Layer or Transfer Layer, while the testing of flight command hardware/software requires data entry at the Coding and Physical Layers. Therefore, a generic implementation of the standard should allow data entry at any layer of the services.
2. Commands to be sent to the spacecraft individually or in groups:
   The services allow you to group commands by putting multiple Packets in one Segment and/or multiple Transfer Frames in one CLTU.
3. Configuration of each layer:
   Each layer of the services provides a number of user-definable parameters, which must be set to match the flight layers used on your particular spacecraft.

Interface Requirements

Fig. 1 Command Data Flow

As described above, to send commands to the spacecraft, you send messages to the Telecommand Services.

The Telecommand Services convert the given data to a form the spacecraft can understand, and the resulting data is transmitted to the spacecraft.

The format of these messages must support communication of the following information:

- The command data you want sent and what form it is in (e.g. packet, segment)
- How you want the data grouped (e.g. multiple packets in one segment)
- How each layer of the Telecommand Services should be configured to speak to your spacecraft.

Interface Goals

An important goal is to provide a simple interface to the Telecommand Services. This is challenging because there are many forms of data to communicate.

The message format is your interface to the Telecommand Services. Thus, it should meet some of the same goals any good user interface would. Namely:

- Messages should be easy to generate and read;
- All messages should have a consistent format, and be self-defining;
- Messages should be available for all functions that you may need to perform;
- Messages should indicate the action to be taken and contain the data to take it upon;
- The system should be expandable, meaning additional message types can be added without modifying existing functions and formats;
- Feedback should be given to each message indicating success or failure.

The solution–concepts

Interface concepts (objects)

We chose to base our message format on another CCSDS standard, the Standard Formatted Data Unit (SFDU). Some characteristics of SFDUs:

1. There are several classes of SFDUs, including I-class (we'll call these envelopes) and Z-class (we'll call these mailbags).
2. Envelopes always contain data.
3. Mailbags may contain envelope(s) and/or other mailbag(s).
4. Each mailbag/envelope is labeled with a four character string describing it's contents.

Our messages each contain one or more objects; each object is an SFDU. Some characteristics of these objects are:

1. There are 2 types of objects: envelopes and mailbags.
2. An envelope contains one piece of data to be acted upon.
3. A mailbag contains a group of envelopes and/or mailbags, (i.e. mailbags are used to combine multiple input data items into a single output data item)
4. The label describes the contents of an object (e.g. whether it is a packet, segment, or whatever), and specifies the actions to be taken with the data.

Implementation concepts (message conversion)

When you send a message to the Telecommand Services, it is passed through each layer. When a layer receives a message, it applies the following rules to each mailbag and envelope within the message and then returns the resulting message.

1. If the label on an envelope indicates input data for this layer, then the envelope is replaced with an envelope containing output data from this layer (built from the contents of the input envelope). For example, the Segmentation Layer replaces each Command Packet envelope with a Command Segment envelope.
2. If the label on a mailbag indicates grouping to be performed by this layer, then the mailbag is replaced with an envelope containing output data from this layer (built from the contents of the input mailbag). For example, the Segmentation Layer replaces a Segment mailbag with a Segment envelope (all the packets within the mailbag are combined into one Segment).
3. If the label on an envelope indicates a Configuration Directive for this layer, then the envelope is removed and the directive is executed.
4. Any mailbag/envelope whose label is unrecognized is left in the message. This is a key characteristic of the implementation because it ensures that objects (i.e. mailbags and envelopes) pass through the services until they reach the applicable layer. It also ensures that the existing implementation is not affected when new objects are defined. Note: All envelopes are checked, including those within unrecognized mailbags.

After the message has passed through each layer, the data inside will have been transformed appropriately to a form that your spacecraft can understand. The resulting data is then transmitted to your spacecraft.

The solution–specifics

List of objects

This section lists the objects that a Command Source may include in messages sent to our Telecommand Services. The following are defined:

Table 2 Envelope Objects

Label	Meaning
DSEG	configuration directive for the Segmentation Layer.
DXFR	configuration directive for the Transfer Layer.
DCOD	configuration directive for the Coding Layer.
DPHY	configuration directive for the Physical Layer.
CPKT	Command Packet data
CSEG	Command Segment data
CFRM	Command Transfer Frame data
CLTU	Command Link Transmission Unit data
CSPA	Command data for the spacecraft

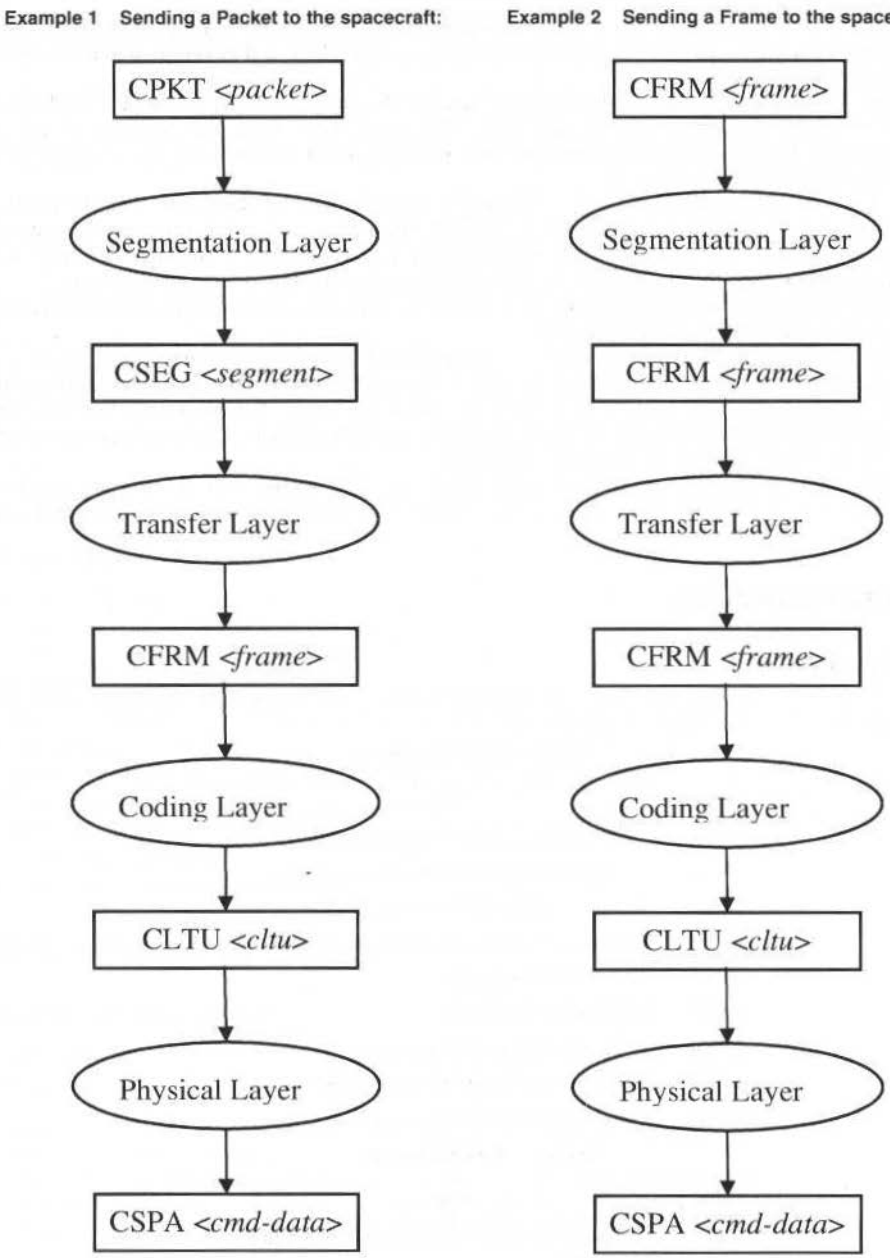
Table 3 Mailbag Objects

Label	Meaning
CSEG	build one Command Segment from all the data in this mailbag.
CLTU	build one CLTU from all the data in this mailbag.

Commanding Examples:

A) Data entry at each layer of the services:

Envelopes containing command data are passed through the Telecommand Services until they reach the applicable layer; from then on they get converted.


Example 1 Sending a Packet to the spacecraft:
CPKT <packet>
Segmentation Layer
CSEG
Transfer Layer
CFRM <frame>
Coding Layer
CLTU <cltu>
Physical Layer
CSPA <cmd-data>
Example 2 Sending a Frame to the spacecraft:
CFRM <frame>
Segmentation Layer
CFRM <frame>
Transfer Layer
CFRM <frame>
Coding Layer
CLTU <cltu>
Physical Layer
CSPA <cmd-data>

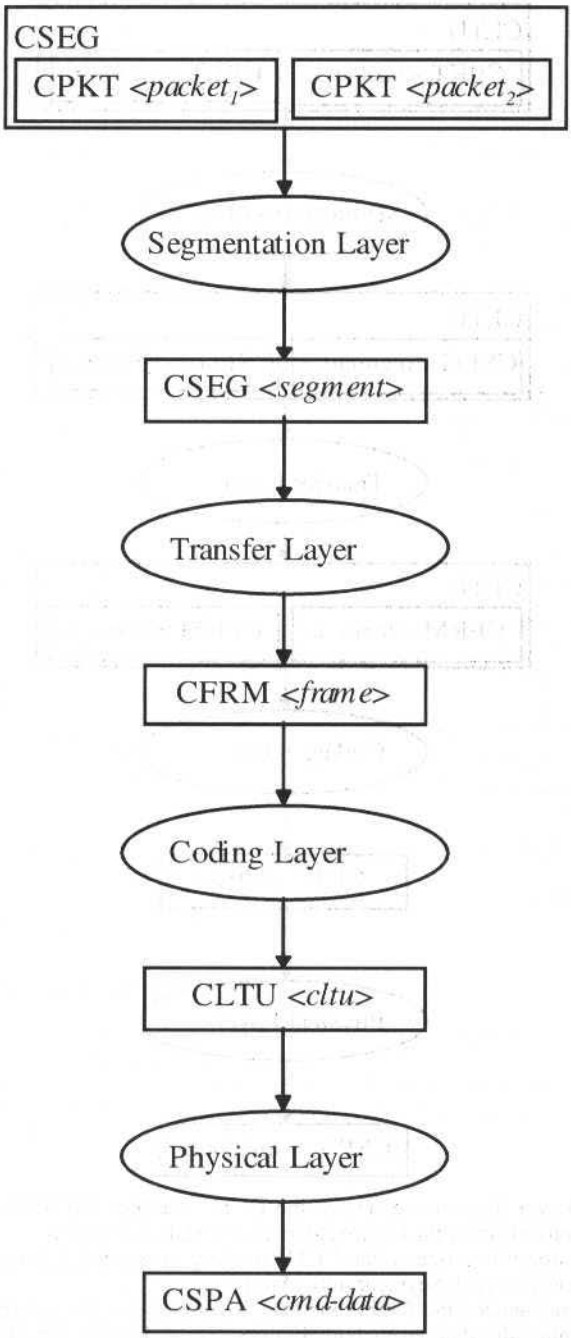
The Packet envelope is recognized at the Segmentation Layer, so headers/trailers are added by every layer.

The Frame envelope is not recognized until it reaches the Coding Layer, so headers/trailers are added beginning with the Coding Layer. Note that the Segmentation and Transfer Layers do not modify the message at all.

B) Data aggregation:

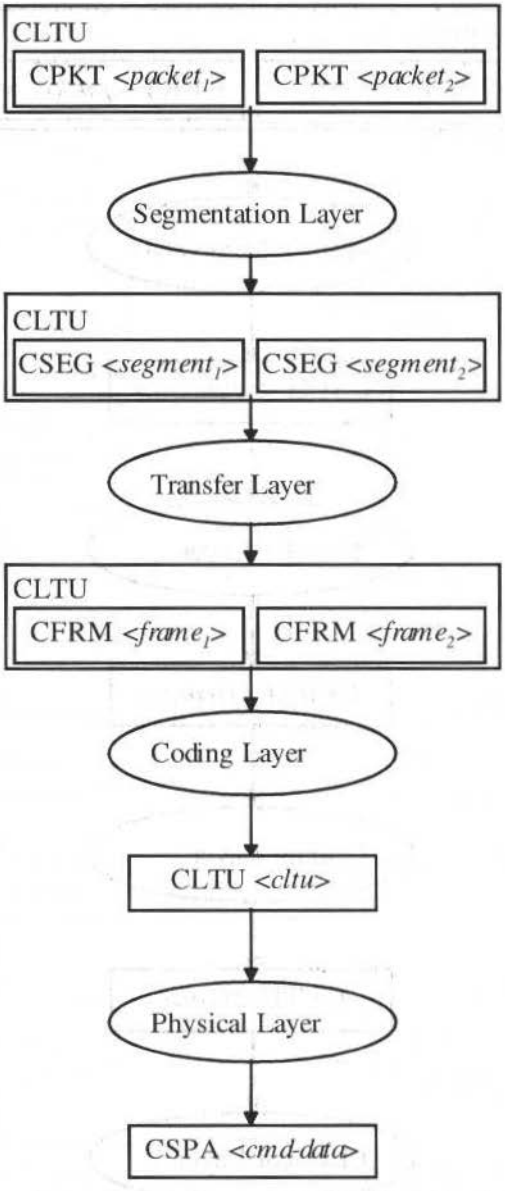
Data aggregation is requested by putting envelopes in a mailbag. The mailbag remains in the message until it reaches the applicable layer; envelopes within mailbags are converted in the normal way by each layer.

Example 3– 2 Packets into 1 Segment:


CSEG
CPKT <packet1>
CPKT <packet2>
Segmentation Layer
CSEG
Transfer Layer
CFRM <frame>
Coding Layer
CLTU <cltu>
Physical Layer
CSPA <cmd-data>

The Segmentation Layer recognizes the Segment mailbag, so it replaces the entire mailbag with a Segment envelope containing the data from both command packets (i.e. both packets are combined into 1 Segment). Headers/trailers are added in the usual way by the remaining layers.

Example 4– 2 Transfer Frames into 1 CLTU:



The Segmentation Layer does not recognize the CLTU mailbag, but it does recognize the Packet envelopes inside, so it converts each Packet envelope to a Segment envelope.

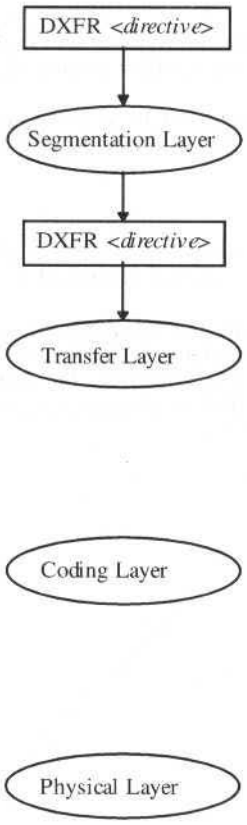
The Transfer Layer doesn't recognize the CLTU mailbag either, but it does recognize the Segment envelopes inside, so it converts each Segment into a Frame.

The Coding Layer recognizes the CLTU mailbag, so it replaces the entire mailbag with a single CLTU envelope containing the data from both Frames (i.e. 2 Frames are combined into 1 CLTU). Headers/trailers are added in the usual way by the remaining layer.

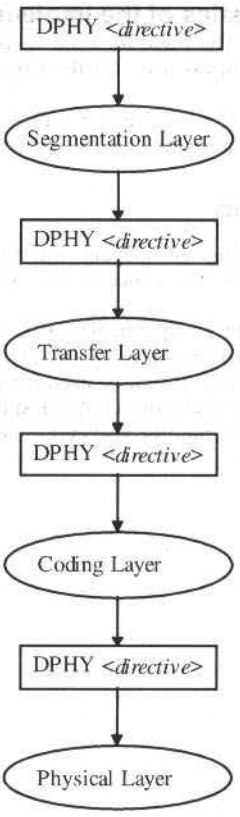
C) Configuration of each layer:

Envelopes containing configuration directives are passed through the Telecommand Services until they reach the applicable layer; at that point they get executed and removed.

Example 5 - A configuration directive for the Transfer Layer



Example 6 - A configuration directive for the Physical Layer



The Transfer-layer directive envelope is not recognized until it reaches the Transfer Layer, where the directive is executed and the envelope is removed.

The Physical-layer directive envelope is not recognized until it reaches the Physical Layer, where the directive is executed and the envelope is removed.

Responses to the Command Source:

When you send messages to the services, you receive responses indicating:

1. if your message was accepted or rejected, and
2. whether or not your command(s) were successfully delivered to the spacecraft.

In writing this paper, we felt the concepts were more clearly illustrated if the response mechanism is not included.

The solution–characteristics

Characteristics of the interface

Flexible: A single interface can handle any valid CCSDS commanding model.

Object-oriented: Each message from the command source specifies both the input data (e.g. Command Packets) and the actions to be performed with the data (e.g. to combine multiple packets into one segment).

Expandable: New capabilities are added by defining new objects (SFDUs). Adding new objects does not affect the existing implementation, because objects pass through the services until they reach the

applicable layer. This is important, since additional layers (outside the Telecommand Services) are needed to communicate with different Ground Stations.

Characteristics of the implementation

Modular: The logic for each service layer is contained in a separate Application Program Interface (API). The objects used for the Command Source interface are also used for communication between layers.

Distributable: The design can be split along any layer boundary for distribution across multiple platforms. Because the objects represent data in ASCII format, there is data consistency across platforms.

Conclusion

By choosing an elegant interface, this complex problem was solved with a straightforward implementation. The concepts we used in this interface are general enough to be used in solving other problems.

The resulting implementation is currently being used by NASA (National Aeronautics and Space Administration) for integration & test of the Microwave Anisotropy Probe (MAP) and Earth Observer-1 (EO-1) spacecraft. It will be used for post-launch operations of these spacecraft as well as the Imager for Magnetopause to Aurora Global Exploration (IMAGE) spacecraft. Use of the same implementation and interface across multiple spacecraft has yielded significant cost savings.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp.318-324

ISSN 0100-7386
Printed in Brazil

Autonomous Command Operations of the Wire Spacecraft

Mike Prior
Keith Walyus
Richard Saylor
Goddard Space Flight Center
20771, Greenbelt, MD. U.S.A.
michael.w.prior@gsfc.nasa.gov
keith.d.walyus@gsfc.nasa.gov
rick.s.saylor@gsfc.nasa.gov

Abstract

This paper presents the end-to-end design architecture for an autonomous commanding capability to be used on the Wide Field Infrared Explorer (WIRE) mission for the uplink of command loads during unattended station contacts. The WIRE mission is the fifth and final mission of NASA's Goddard Space Flight Center Small Explorer (SMEX) series to be launched in March of 1999. Its primary mission is the targeting of deep space fields using an ultra-cooled infrared telescope. Due to its mission design WIRE command loads are large (approximately 40 Kbytes per 24 hours) and must be performed daily. To reduce the cost of mission operations support that would be required in order to uplink command loads, the WIRE Flight Operations Team has implemented an autonomous command loading capability. This capability allows completely unattended operations over a typical two-day weekend period. The key factors driving design and implementation of this capability were:

1. *Integration with already existing ground system autonomous capabilities and systems,*
2. *The desire to evolve autonomous operations capabilities based upon previous SMEX operations experience – specifically the SWAS mission,*
3. *Integration with ground station operations both autonomous and man-tended,*
4. *Low cost and quick implementation, and*
5. *End-to-end system robustness.*

A trade-off study was performed to examine these factors in light of the low-cost, higher-risk SMEX mission philosophy. The study concluded that a STOL (Spacecraft Test and Operations Language) based script, highly integrated with other scripts used to perform autonomous operations, was best suited given the budget and goals of the mission. Each of these factors is discussed in addition to use of the SWAS mission as a testbed for autonomous commanding prior to implementation on WIRE. The capabilities implemented on the WIRE mission are an example of a low-cost, robust, and efficient method for autonomous command loading when implemented with other autonomous features of the ground system. They can be used as a design and implementation template by other missions interested in evolving toward autonomous and lower cost operations.

Keywords: Automation, satellite operations, low-cost operations

Introduction

The SMEX missions have historically been leaders for new operational and architectural ground system philosophies at the Goddard Space Flight Center. The fundamental philosophy for the SMEX Flight Operations Team is the multi-mission operations concept. This concept was developed by the FOT, which had one team responsible for all of the SMEX spacecraft. These spacecraft currently include: SAMPEX, FAST, TRACE, and SWAS , which have all already launched and are operational. Also included in the SMEX series is WIRE , which is scheduled to launch in February of 1999. The multi-mission concept is effective because the SMEX FOT is involved in all aspects of a mission development and on-orbit operations. A portion of the team works in the pre-mission phase of a mission that includes integration and testing and developing the operational concept for a mission. Beside their main tasks in the pre-mission phase, a secondary task is to transfer the knowledge back to the core team that will eventually take on the routine operation responsibilities.

SMEX Operational history

SAMPEX used the Transportable Payload Operational Control Center for its ground system, and was the first major GSFC mission to migrate away from a traditional mainframe ground system. TPOCC used a workstation-based system that provided the mission with a high degree of flexibility for the time. Data servers with additional workstation strings could be added for operational support during high activity periods, such as the launch and early orbit phase. The operational philosophy still reflected a traditional operational approach whereas the flight team and the I&T teams were two separate teams and different

ground systems were used for I&T and ops. At launch, the ops team still supported seven days a week, 24 hours a day. Although the ground system architecture was beginning to evolve, little improvement had been made to the operational concept.

After the launch of SAMPEX in July 1992, the SMEX FOT developed the first step in automating operations and reducing costs for the mission. The concept of "blind passes" was created and tested. The concept allowed the SMEX FOT to staff the control center seven days a week, but only 16 hours a day. During the remaining eight hours, the spacecraft was commanded from the stored command load. The on-board command load allowed the spacecraft to dump its science data to the ground during the supports without any intervention from the ground. The FOT configured the spacecraft to dump the data three minutes after AOS, which allowed the ground station personnel time to account for station masking or acquisition problems. After this concept was tested, both the SMEX FOT and NASA gained confidence, and staffing was further reduced to 12 hours a day, seven days a week.

Because of problems with the launch vehicle, FAST launch operations were twice ordered to stand down. After the second FAST stand-down in October 1995, a 25% FOT staff reduction was needed to reduce the overall budget for SMEX. The team's staff was reduced from 20 people to 15. When the reduction occurred, SMEX still only had one spacecraft in orbit, but the reduction forced the team to examine the routine operations work and eliminate or reduce the amount of work that was required daily.

Although the ground system architecture remained relatively similar to the SAMPEX mission, the FAST team began to incorporate new and more effective operational strategies. The first change started in the pre-mission phase when the ops team was integrated into the I&T team. Although an integrated team is standard practice for most missions today, at the time it was a new direction for GSFC missions. Changes also were incorporated into the ops concept. Once FAST launched in August 1996, the SMEX team began multi-mission operations. Because of the use of blind passes, and with FAST transitioning to routine operations, the control center staffing was able to be reduced from 12 hours a day to 8 hours a day. In October 1997, the SMEX FOT started five-day operations. The control center was only staffed Monday through Friday, 8 hours each day. To meet this level of reduced staff coverage, several other tools had to be incorporated. The use of blind passes was still being used during the SMEX FOT off-hours.

One of the problems in going to five-day operations was preventing the spacecraft watchdog timer from timing out. On all SMEX spacecraft, a watchdog timer monitors the time receipt between ground commands. If this time between commands exceeds 24 to 28 hours (dependent on which mission), the spacecraft will re-boot itself causing an impact to the science mission and requiring one to several days to fully recover the satellite. A method was developed that allowed the ground station personnel at Wallops Flight Facility to uplink no-operation commands from a file during normal supports when the SMEX FOT was not staffing the control center. The no-operation commands were uplinked on every blind pass that Wallops ground station supported. This procedure proved effective and reliable. Since the start of five days operations, none of the spacecraft watchdog timers have ever caused a spacecraft re-boot.

A second problem that was encountered for five-day operations was weekend on-board command loads. The on-board command load had to be increased from 24-hour coverage to 48-hour or 72 hour coverage to preclude the need for uplinking command loads to the spacecraft on Saturday or Sunday. The SMEX FOT looked at several activities and command sequences in the stored command load and was able to reduce or eliminate some of the commands in the loads. This reduction allowed the on-board command loads to cover 48 or 72 hours and two loads were uplinked to control the spacecraft over the weekend. To reduce the risk to the spacecraft and prevent large amounts of data loss, other elements in the end-to-end data flow were augmented. Arrangements were made with the ground stations and data processing teams to contact the flight ops team if a problem from a pre-defined list occurred. Since the start of the five-day operations, science data capture has not declined and remains above 99% for all SMEX missions and no spacecraft anomalies have caused any serious problems because of the reduced staff coverage.

With the launch of TRACE in April 1998, the FOT incorporated the same five-day operation concept for its normal operations. The TRACE team added new tools for monitoring the spacecraft. The SERS was created by NASA to help monitor spacecraft. SERS scans through the system event logs looking for anomalous events or violations or any other messages that the flight operations define. If an anomalous event or violation is found, the system will notify the SMEX FOT by email and pages. A list of team members is notified until an acknowledgement is received back to SERS from the pager. This system allowed the SMEX FOT to gain more confidence and insight into the health and safety of the spacecraft during off-hours. Because of the success of the system SWAS has started to use it after launching in December 1998. WIRE is currently being configured on the system and FAST will start using the system in early 1999.

The ground system also underwent significant architectural changes. For SAMPEX and FAST, the ground system was based on workstations with separate front-end processors. TRACE used the ITOS

system for its ground system. This system incorporated the front-end processor functions into the workstation thus requiring only one unit. The ITOS ground system also was the main ground system for all of SMEX I&T. Therefore the FOT developed many of the pages and procedures during the I&T phase, which allowed the FOT to re-use most of them for operations. Additionally, the ITOS system was hosted onto desktop and laptop PC's in addition to workstations for added flexibility. Although workstations were used primarily for the TRACE control center, much less expensive desktop and laptop PC's could now be used for high activity periods for added cost savings. The FOT used the laptop PC's for procedure and documentation development in their office or even at home, and then easily integrated the changes into the system.

Unique problems arose with the SWAS and WIRE operations plans. The time-lines in the on-board command load are numerous because it gives several targets for the spacecraft to investigate. Because of their large size, the command loads time period could not be extended to cover the entire weekend. For these two missions to meet the current five-day operations concept and to maintain the same level of science data collection, the SMEX FOT developed and created an autonomous command load procedure that uses STOL language and UNIX scripts. The ground system architecture and spacecraft command concepts used for autonomous commanding will be discussed in the following sections.

Autonomous Architecture Overview

The current capabilities developed for autonomous operations are in the support of autonomous station contacts, which are referred to as blind passes. These are performed using virtually the same processes as tended supports except that automation tools perform system configurations, receipt and monitoring of data, and post-pass processing of pre-recorded housekeeping and science data in place of flight team personnel. During all normal contacts the WIRE Spacecraft autonomously transmits real-time housekeeping telemetry, dumps stored housekeeping , and science telemetry data. Normal contacts are pre-scheduled contact times of approximately 8-10 minutes duration and occur 3 times daily. All telemetry management, transmitter ON/OFF, and other spacecraft management functions related to supporting normal contacts are performed via onboard stored commanding (i.e. a command load or ATS load).

The ITOS ground system Mission Operations Center or MOC autonomously performs self configurations to support the contact as would be done by flight team personnel, and configures TCP/IP and UDP/IP network connections as a server. The ground station initiates port connections as the client. The connections are normally performed manually by station personnel but may also be performed autonomously if the station is also un-tended. ITOS receives real-time data during the contact and stored housekeeping data post-contact via FTP. Recorded science data is FTP'd directly to the science team post-contact. The MOC processes both real-time and post-contact data to look for anomalous spacecraft conditions. Flight team members are notified via pager of any such conditions.

The current capabilities, as described in the above overview, have been further developed to also provide for autonomous commanding of the spacecraft. The following sections describe the basic tools of the ground system that are used to provide this capability and how they have been adapted to support command loading for the WIRE mission.

Key Components: ITOS, SERS, UNIX Scripts

The core tools and features of the MOC needed to support autonomous operations consist of the ITOS, SERS (Spacecraft Emergency Response System), and Unix/Solaris scripts. The role of each is discussed below.

ITOS

ITOS is a workstation based software package developed in-house at GSFC that runs under the SUN/Solaris operating system. ITOS provides many features but its key capabilities for the support of autonomous operations are:

- STOL (System Test and Operations Language) procedural scripting
- Limit and configuration monitoring of telemetry
- Event capturing and logging
- Telemetry archival and replay

Time based execution of STOL procedures drive the configuration and setup for station contacts, the real-time data processing and commanding during a contact, and the post-pass processing of stored housekeeping data. The procedures used to support autonomous operations (i.e. the blind pass

procedures) are highly enhanced variations of scripts used to perform the same basic functions during a flight team tended contact.

ITOS provides a limit and configuration monitoring capability of spacecraft telemetry that is configured to create 'event messages' whenever a telemetry point is not within an expected range or not set to an expected discrete value. Other messages are generated from ITOS system events, the status of telemetry and command processing, and the execution of STOL and Unix scripts. All such events are written to an event log and are created for both real-time and post-pass playback data.

SERS

The SERS system was developed out of a general drive to reduce operations costs by reducing flight team personnel support requirements. It was first implemented on the TRACE mission and will be used on all other SMEX missions except SAMPEX. The SERS system is instrumental for ensuring automation robustness. It notifies flight team personnel of the occurrence of any anomalous conditions including the failure of the STOL scripts themselves to complete command loads and other activities. It performs this function via processing of event logs (real-time and post-pass) created from the contact. Each log file is opened and read for limit, configuration, and ITOS system event messages that are considered anomalous. All such messages are then transmitted to several flight team personnel, who are designated as being on call during the support. The flight team member receiving the SERS alert must use his/her pager to acknowledge receipt or the system will alert the next member in the list. All event messages appear on the pager display in full format and length as they would be read from the workstation console. After reviewing the message(s) the FOT member can then decide what action, if any is necessary.

Unix/Solaris Scripts

Unix scripts are used to perform various cleanup activities including moving files between workstations and within directories of a workstation. Examples of this include the automatic transfer of event log files from the prime and backup workstations to the SERS, the transfer of attitude and orbit files (created from post-pass playback of housekeeping data) to the flight dynamics support system, and the transfer of housekeeping data to the data processing system. As data is moved to its final destination and/or archival location(s) it is deleted from intermediate storage directories. All ground system elements are networked and all file transfers are performed via FTP. A key UNIX script was developed for the Auto-Commanding procedure that monitors the ITOS software. If the software processes gets killed or hung, the script will perform a system reboot and re-configure the software. This allows the flight ops team to have confidence that science data loss or a failure in the uplink of a store command load will not be due to mission software failure on ITOS.

Spacecraft Command Concepts

Multi-layered command protocol

To effectively perform autonomous spacecraft commanding, ITOS STOL procedures were modified to include a thorough verification of successful load completion. This verification is based upon the combined spacecraft and ground system command verification protocol.

Commanding and command verification for the WIRE spacecraft are identical to the other SMEX spacecraft and relies upon a multi-layer command protocol based upon CCSDS COP-1. The ITOS system implements the FOP and the spacecraft implements the FARM in its command ingest software. In addition, the system performs command verification via a software task monitoring the command counter and end-item verification. End-item verification is accomplished by monitoring the telemetry for spacecraft configuration changes. For both tended and autonomous operations, the STOL procedures make extensive use of both software command counter checks and end-item verification. The COP-1 command verification is automatically performed in the background for all commands. The successful or unsuccessful history of all commanding is reported in the ITOS event log.

Command Load Uplink

The WIRE mission requires that two command loads be uplinked to the spacecraft daily. Each load contains spacecraft management, instrument configuration, and spacecraft maneuver commands covering a twelve-hour duration with approximate size of 18 Kbytes. The two loads together span 24 hours of operation. Uplink for the command load set is performed during a single station contact of 8-10 minute duration and requires 4-6 minutes of uplink time. The load set is uplinked approximately 18 hours prior to the current set expiring.

- How much of the former in-MOC activities can be automated, and
- How well the alert system works.

NASA/GSFC's Advanced Architectures and Autonomy Branch (Code 588) is working to build technologies to help facilitate the transition to lights-out operations. One such effort is a research project called the Virtual Mission Operations Center (VMOC). The goal of the VMOC project is to work closely with operations staff to develop new technologies and workgroup computing concepts to more effectively meet the new objectives for mission operations. The VMOC project is exploring the feasibility and effectiveness of technologies such as computer-supported collaborative work (groupware), wireless communications devices, and distributed operations in order to meet these goals.

One result of the VMOC effort is the development of the Spacecraft Emergency Response System (SERS). SERS is a Web-based suite of tools created to monitor and summarize spacecraft operations and notify the SCT when anomalous conditions arise. During operations, SERS provides autonomous logging of events that require human attention (as defined by the SCT). SERS also automatically generates summaries of each pass. In addition, SERS supports automatic schedule changes and the manual reporting of anomalies. Upon logging anomalies, SERS' software agents perform customizable workflow processes that automatically generate reports and alert the appropriate SCT member(s) and/or engineer(s). This paper provides more detail on SERS' reporting capabilities.

Automation During Operations

While SERS supports anomaly logging and tracking during I&T, its capabilities are best utilized during the operational phase of a spacecraft. During routine operations, SERS receives e-mail notifications from expert systems and automatically examines ground system log files for anomalous conditions. When such a condition is detected, SERS creates an Event Report to document the condition(s) and automatically notifies the appropriate on-call team members. SERS also autonomously generates Pass Summary Reports that summarize the contents of each ground system log file that it examines. In addition, SERS handles schedule changes submitted by the Deep Space Network (DSN).

Many ground systems generate log files for each contact between the ground station and a spacecraft. These contacts occur throughout the day; so in lights-out operations, many of these contacts are performed autonomously. The log file that is generated contains documentation on the status of the spacecraft and its subsystems (i.e., health and safety data), along with any commands or procedures initiated by the SCT. SERS examines these log files with two different processes (1) to determine if a condition has arisen that requires the SCT to be notified and (2) to summarize notable messages and errors that were detected during the pass. These two processes are independently configured and executed, so that changes in the configuration of one will not affect the processing of the other.

Event Reports & Notifications

In SERS, an "event" is defined as a condition that merits the attention of the SCT and that may require some human intervention. When a new event occurs, SERS determines whom to notify based upon the event's characteristics, user-defined filters, an on-line schedule, and profiles of the SCT. SERS then mediates the response to the alert by notifying back-up personnel if the initial individuals notified of the problem do not respond to the alert. SERS uses a variety of mechanisms (pagers, e-mail, Web, and telephone) to communicate with the SCT. More detailed accounts of how SERS performs these functions are described in Fox, et. al. [13] and Baker, et al.[14] A high-level diagram of the process is shown in Fig. 1.

Events in SERS are generated in two ways: (1) by receipt of a formatted e-mail message sent from an expert system or another outside process and (2) by the SERS Regular Event Processor (REP). In both cases, SERS receives the details of the event in a formatted e-mail message and autonomously generates a Pass Event Report, which is available immediately via the Web. By providing Web access, SERS allows the SCT to view reports from any computer that has a Web browser. A typical Pass Event Report is shown in Fig. 2. SCT personnel who are notified of the problem can access the Web page to obtain complete information about the event.

If the spacecraft's front-end is an expert system, like Altairis,[15] the expert system sends SERS an e-mail message depicting the state of the spacecraft when it determines that an anomalous condition has arisen. In addition, other external processes can automatically e- mail SERS when problems are detected. These currently include weather alerts, along with various processes that monitor the status of the other system hardware and software. Each alert causes a Pass Event Report to be generated and designated SCT personnel to be notified.

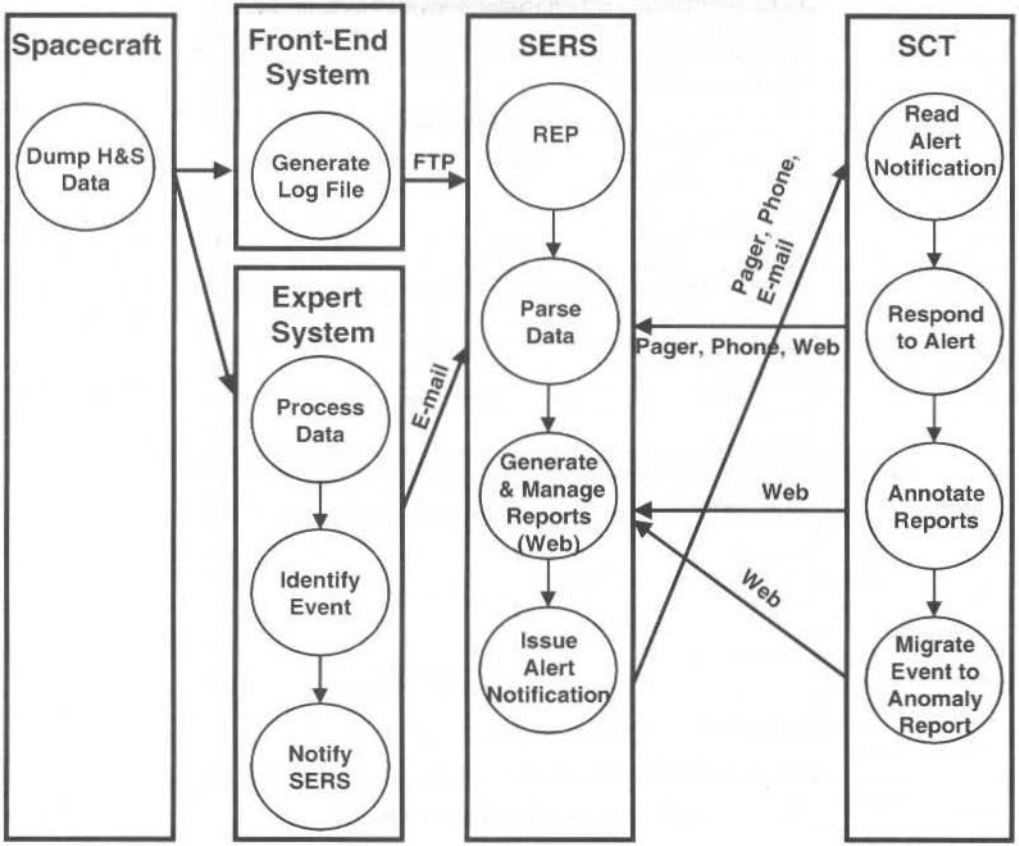


Fig. 1 SERS-Mediated Workflow for Operational Event Handling

If a mission is not using an expert system as part of its front-end, the SERS REP parses the log files from ground systems, like ITOS[16] and ASIST,[17] to determine if they contain any conditions requiring SCT notification. The SCT defines the items of interest over the Web by completing forms that define the elements of interest in the log file. These are known as filters. The SCT then defines the criteria for those filters that will either trigger an alert, provide contextual information, or inhibit an alert notification.

When SERS finds the filter criteria in a log file, it automatically generates a Pass Event Report that includes information about the identified problem and any pre-selected contextual data. Alternatively, inhibit filters can be defined to prevent unnecessary alert notifications from being sent. Inhibit filters are generally narrowly defined to match certain recurring messages that may contain data that are not nominal, but are nonetheless not of interest to the SCT. This reduces the number of unnecessary notifications to a minimum.

Pass Summaries

SERS generates Pass Summary Reports based on data from two sources: (1) the Deep Space Network (DSN) schedule and (2) the ground system log file. SERS first uses DSN's 7-day pass plan, or schedule, to automatically generate skeleton reports containing schedule data for each pass for the spacecraft it monitors. A typical DSN Schedule for SMEX missions is shown in Fig. 3. As the schedule is revised throughout the week, these skeletal Pass Summary

Table 1 Test Matrix

No.	Test Description:	Status:
1	Conduct procedure tests to eliminate procedure errors.	Complete
2	Convert procedures and scripts to operate for SWAS	Complete
3	Monitor procedure in real-time using the SWAS spacecraft	Open
4	Test and verify possible failure scenarios while monitoring the procedure in real-time	Open
5	Execute the procedure with real-time monitoring for SWAS	Open
6	Apply any enhancements and updates to the WIRE procedures and scripts.	Open

Summary

Because of the automation tools that the SMEX FOT incorporated into normal operations of SMEX spacecraft, the team size was reduced by approximately half from the original multi-mission team size. These reductions resulted in cost savings to every SMEX mission and especially helped those missions remain funded in their extended mission phase. All of the techniques that were implemented were done with common tools that are available to every control center. The only specialized platform that SMEX currently uses is the SERS system. The remainder of the automation is accomplished through UNIX scripts and STOL procedure that were developed by the SMEX FOT. The team is continuously looking at ways to further automate operations, not only to reduce cost, but also to reduce the risk of errors. The only area that has not been seriously examined for automation is the off-line analysis that each mission is required to perform. Some initial work has been done in this area, but the work is still labor intensive.

Abbreviations

AOS: Acquisition of Signal;
ATS: Automatic Time Sequence;
CCSDS: Consultative Committee for Space Data Systems;
COP: Command Operation Procedure;
FARM: Frame Acceptance Reporting Mechanism;
FAST: Fast Auroral Snapshot Telemeter;
FOP: Frame Operation Procedure;
FOT: Flight Operations Team;
FTP: File Transfer Protocol;
GSFC: Goddard Space Flight Center;
I&T: Integration and Test;
ITOS: The Integrated Test and Operations System;
LOS: Loss of Signal;
MOC: Mission Operations Center;
NASA: National Aeronautics and Space Administration;
PC: Personal Computer;
SAMPEX: Solar Anomalous Particle Explorer;
SERS: Spacecraft Emergency Response System;
SMEX: Small Explorer;
STOL: Spacecraft Test and Operations Language;
SWAS: Sub-Millimeter Wave Astronomy Satellite;
TCP/IP: Transmission Control Protocol/Internet Protocol;
TPOCC: Transportable Payload Operations Control Center;
TRACE: Transition Region and Coronal Explorer;
UDP/IP: Uniform Data Packet Internet Protocol;
WIRE: Wide Angle Infrared Explorer.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp.325-331
ISSN 0100-7386
Printed in Brazil

WEB-Based Automated Reporting: Saving Time, Money and Trees

Jeffrey A. Fox[1]
Pacific Northwest National Laboratory
901 D Street, SW, Suite 900
DC 20024, Washington
jeff.fox@pnl.gov

Cindy Starr
Paul Baker
Kai-Dee Chu
Global Science and Technology, Inc.
6411 Ivy Lane Suite 300
Greenbelt, Maryland 20770
pbaker@gst.com
chu@gst.com
starr@gst.com

Julie Breed
NASA/Goddard Space Flight Center
Greenbelt Road
Greenbelt, Maryland 20771
julie.breed@gsfc.nasa.gov

Mick Baitinger
8403 Colesville Road, Suite 915
Silver Spring, Maryland 20910
NEXGEN Solutions, Inc.
mbating@ngsinc.com

Abstract

In the current "smaller, faster, cheaper" world of space programs, there is increased pressure to dramatically reduce costs. This has led to the design of smaller spacecraft and to a reduction in mission personnel. One of the major obstacles in reducing staff in all phases of a mission is the significant amount of paperwork that the staff must complete. For example, during routine operations, the Spacecraft Control Team (SCT) may need to complete a variety of reports including anomaly reports, pass summaries, and weekly summary reports. Boxes of these reports are archived indefinitely. The Spacecraft Emergency Response System (SERS) is an innovative Web-based suite of tools created to support "lights-out" operations. SERS automates many of the reporting activities required during both operations and integration and test (I&T). Currently, SERS is being used by several Small Explorer (SMEX) missions. In addition, SERS will be used by the Hubble Space Telescope (HST), and missions from Middle Explorers (MIDEX) and the New Millennium Program (NMP).
Keywords: *Lights-out operations, Automation, SERS, Anomaly management.*

Introduction

At NASA's Goddard Space Flight Center (GSFC) and at space centers elsewhere, the trend is to reduce mission operations costs through automation, as demonstrated by a number of papers presented at the 1998 SpaceOps Conference,.[1-10] Some organizations are even moving to lights-out operations for unmanned spacecraft,[11-14] where many of the standard operational activities are automated. In this operational setting, the spacecraft control team (SCT) no longer continuously occupies the mission operations center (MOC). The goal of lights-out operations is to reduce the overall cost of spacecraft operations by reducing the total number of hours that personnel are directly involved in day-to-day activities. Instead, a reduced number of team members work a standard shift (perhaps 9-5, Monday through Friday) performing those tasks that cannot be automated in a cost-effective fashion. The rest of the time, the SCT acts as on-call team members who respond to notifications of anomalous conditions.

The effectiveness of this approach depends on a number of factors, including:

- How well anomalous conditions can be identified

---

[1] Pacific Northwest National Laboratory is operated for the U.S. Department of Energy by Battelle under Contract DE-AC06-76RLO 1830.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp.325-331

ISSN 0100-7386
Printed in Brazil

WEB-Based Automated Reporting: Saving Time, Money and Trees

Jeffrey A. Fox[1]
Pacific Northwest National Laboratory
901 D Street, SW, Suite 900
DC 20024, Washington
jeff.fox@pnl.gov

Cindy Starr
Paul Baker
Kai-Dee Chu
Global Science and Technology, Inc.
6411 Ivy Lane Suite 300
Greenbelt, Maryland 20770
pbaker@gst.com
chu@gst.com
starr@gst.com

Julie Breed
NASA/Goddard Space Flight Center
Greenbelt Road
Greenbelt, Maryland 20771
julie.breed@gsfc.nasa.gov

Mick Baitinger
8403 Colesville Road, Suite 915
Silver Spring, Maryland 20910
NEXGEN Solutions, Inc.
mbating@ngsinc.com

Abstract

In the current "smaller, faster, cheaper" world of space programs, there is increased pressure to dramatically reduce costs. This has led to the design of smaller spacecraft and to a reduction in mission personnel. One of the major obstacles in reducing staff in all phases of a mission is the significant amount of paperwork that the staff must complete. For example, during routine operations, the Spacecraft Control Team (SCT) may need to complete a variety of reports including anomaly reports, pass summaries, and weekly summary reports. Boxes of these reports are archived indefinitely. The Spacecraft Emergency Response System (SERS) is an innovative Web-based suite of tools created to support "lights-out" operations. SERS automates many of the reporting activities required during both operations and integration and test (I&T). Currently, SERS is being used by several Small Explorer (SMEX) missions. In addition, SERS will be used by the Hubble Space Telescope (HST), and missions from Middle Explorers (MIDEX) and the New Millennium Program (NMP).

Keywords: *Lights-out operations, Automation, SERS, Anomaly management.*

Introduction

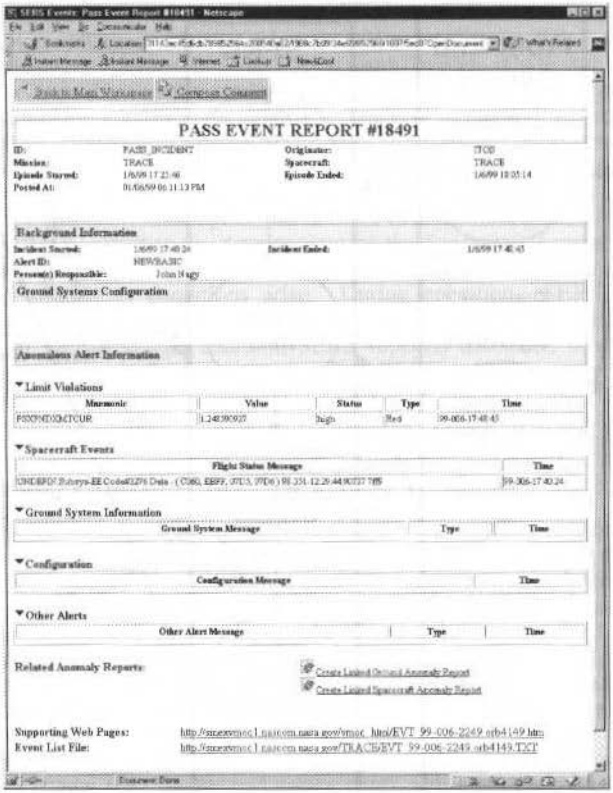
At NASA's Goddard Space Flight Center (GSFC) and at space centers elsewhere, the trend is to reduce mission operations costs through automation, as demonstrated by a number of papers presented at the 1998 SpaceOps Conference,.[1-10] Some organizations are even moving to lights-out operations for unmanned spacecraft,[11-14] where many of the standard operational activities are automated. In this operational setting, the spacecraft control team (SCT) no longer continuously occupies the mission operations center (MOC). The goal of lights-out operations is to reduce the overall cost of spacecraft operations by reducing the total number of hours that personnel are directly involved in day-to-day activities. Instead, a reduced number of team members work a standard shift (perhaps 9-5, Monday through Friday) performing those tasks that cannot be automated in a cost-effective fashion. The rest of the time, the SCT acts as on-call team members who respond to notifications of anomalous conditions.

The effectiveness of this approach depends on a number of factors, including:

- How well anomalous conditions can be identified

---

[1] Pacific Northwest National Laboratory is operated for the U.S. Department of Energy by Battelle under Contract DE-AC06-76RLO 1830.



PASS EVENT REPORT #18491

ID:	PASS_INCIDENT	Originator:	TCO
Mission:	TRACE	Spacecraft:	TRACE
Episode Started:	1/6/99 17:23:46	Episode Ended:	1/6/99 18:05:14
Posted At:	01/06/99 06:11:13 PM		

Background Information

Incident Started:	1/6/99 17:40:24	Incident Ended:	1/6/99 17:48:43
Alert ID:	NEWBASIC		
Person(s) Responsible:	John Nagy		

Ground Systems Configuration

Anomalous Alert Information

Limit Violations

Mnemonic	Value	Status	Type	Time
PSSPNDXACTCUR	1.24839E07	high	Red	99-006-17:48:43

Spacecraft Events

Flight Status Message	Time
UNDEFD? Subsys-EE Code#1276 Data: (C0E0, EBFF, 07C3, 97D6) 98.334.12.29.44.90737 7ff5	99-006-17:40:24

Ground System Information

Ground System Message	Type	Time

Configuration

Configuration Message	Time

Other Alerts

Other Alert Message	Type	Time

Related Anomaly Reports: Create Linked Ground Anomaly Report; Create Linked Spacecraft Anomaly Report

Supporting Web Pages: http://smexvmoc1.nascom.nasa.gov/vmoc_html/EVT_99-006-2249_orb4149.htm

Event List File: http://smexvmoc1.nascom.nasa.gov/TRACE/EVT_99-006-2249_orb4149.TCT

Fig. 2 Typical SERS Pass Event Report

```
*                    WPS 7-DAY OPERATIONS SCHEDULE              VERSION 07 SCHE
*                          CHANGE LOG
*                  WEEK NO. 41 *** 08 Oct 98 - 11 Oct 98
*-------------------------------------------------------------------------------
*DAY START BOT EOT  END   FACILITY USER   ACTIVITY              PASS  CONFIG/ WBK
*                                                              NO.   SOE     CAT
*-------------------------------------------------------------------------------
*----------
*THU 08 Oct
*----------
*----------
*FRI 09 Oct
*----------
*----------
*SAT 10 Oct
*----------
D283 1054 1124 1140 1205  DSS-76   FAST   RTD, 2250K, 2W     8444  xxxxxx- xxx

I283 1110 1140 1210 1245  DSS-29   FAST   RTD, 2250K, 2W     8444  xxxxxx- xxx

I283 1249 1319 1339 1404  DSS-76   FAST   RTD, 2250K, EE     8445  xxxxxx- xxx


*                    WPS 7-DAY OPERATIONS SCHEDULE              VERSION 07 SCHE
*                        ACTIVITIES LISTING
*                  WEEK NO. 41 *** 08 Oct 98 - 11 Oct 98
*-------------------------------------------------------------------------------
*DAY START BOT EOT  END   FACILITY USER   ACTIVITY              PASS  CONFIG/ WBK
*                                                              NO.   SOE     CAT
*-------------------------------------------------------------------------------
*----------
*THU 08 Oct
*----------
 280 2304 2334 0004 0029  DSS-29   FAST   RTD, 2250K, 2W     8417  xxxxxx- KEY

 281 0120 0140 0149 0151  DSS-82   SAM    RTD, 900K, 2W, N   4093  xxxxxx- xxx

 281 0139 0209 0221 0246  DSS-29   FAST   RTD, 2250K, 2W     8418  xxxxxx- xxx

 281 0320 0340 0351 0421  DSS-29   TRAC   RTD, 2250K, 2W     2803  xxxxxx- xxx

 281 0412 0442 0502 0527  DSS-76   FAST   RTD, 2250K, 2W     8419  xxxxxx- xxx

 281 0604 0634 0704 0729  DSS-29   FAST   RTD, 2250K, 2W     8420  xxxxxx- xxx
```

Fig. 3 Typical DSN Schedule for SMEX Missions

Reports are deleted, created, or modified to reflect changes in the schedule. Next, SERS generates a temporary report for each ground system log file once the REP has completed its examination of the file. Information that is determined to be relevant by the SCT is extracted from the log file and stored in a temporary summary. The temporary summary is then matched to the previously generated Pass Summary Report based on date, time, spacecraft, and orbit number. Relevant log file information from the temporary summary is posted automatically into the Pass Summary Report and the temporary summary is subsequently deleted. A typical Pass Summary Report is shown in Fig. 4.

At certain times, matches cannot be automatically determined between the temporary reports and the Pass Summaries generated by the schedule. For example, passes may be taken that are not on the schedule. In this case, the SCT may manually generate an "Unscheduled Acquisition." This report functions in the same way as a Pass Summary Report generated from a schedule. Log file information will automatically propagate the form provided the time/date and orbit number match. Another case is when the information in the log file may not match the Pass Summary Report that was generated from the schedule. This may be due to a data entry error during the pass or a time shift from the original schedule. When this occurs, unmatched log information is posted to the Web. When the unmatched fields are corrected by the SCT, the unmatched log data will post automatically to the correct Pass Summary Report.

Fig. 4 Typical SERS Pass Summary Report (Top Portion)

Schedule Changes

SERS also handles schedule changes from the DSN. As shown in Fig. 5, a person from the DSN can notify SERS of a slip or cancellation of a spacecraft contact via the Web or by calling into SERS with a phone. When using a phone, the user is guided through a series of menus to select (1) the appropriate spacecraft, (2) the contact that must be modified, and (3) the type of modification (e.g., slip or cancel). All menu selections are made via the keypad. Users can enter comments about the modification by speaking into the phone. SERS records the voice as an attachment to the report. SERS then automatically generates a Schedule Modification Report and notifies the appropriate SCT member(s). In the future, SERS will allow on-call SCT members to accept or reject schedule changes via 2-way pager. SERS will then pass the information about the changes on to the front-end expert system so that it may make the appropriate modifications to handle the pass. Similarly, DSN personnel can send more general communications alerts to the SCT via SERS.

VMOC Vision

VMOC's ultimate goal is to provide a suite of integrated collaboration tools to support the design and documentation of the entire life cycle of a spacecraft. In the VMOC program, new longer-range technologies and concepts are being explored. One such concept is an expert system that analyzes the health and safety, or state data, of a spacecraft, as recorded in SERS Pass Event Reports. The system then automatically generates hypertext (Web) links from the SERS report to the appropriate on-line support information. A first step in this process will be to provide tools that increase SERS' utility in diagnosing and resolving anomalies.

For example, if SERS logs limit violations in the attitude control system (ACS) of a spacecraft, the expert system would automatically insert links (as part of the Web report) to schematics of the ACS, documentation on the operation of ACS, descriptions of the out-of-range mnemonics, other anomaly reports similar to that one, and other diagnostic tools.

Conclusion

Moving to a lights-out approach to operations requires the automation of many tasks. SERS is a system designed to facilitate this new paradigm. SERS provides tools to automate report generation for events, passes, and contact schedule changes. It also manages the alert notification and response process for anomalous events. In addition to the capabilities described in this paper, new features are being investigated as part of the VMOC effort and will be integrated into SERS, as they become available.

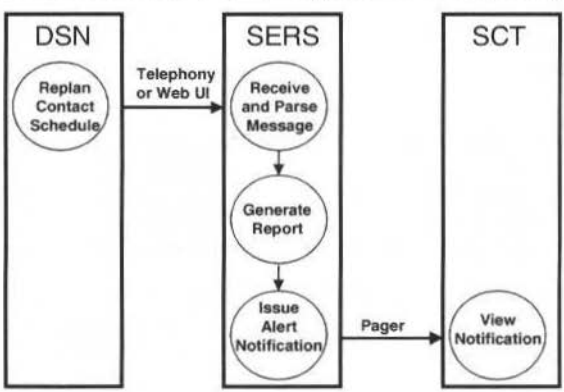


Fig. 5 SERS-Mediated DSN Notification

References

[1]Asoh, D., Sarai, H., and Kogure, S., 1998, "Feasibility Study on Low Cost Autonomous Ground Terminals and Tracking Network". Proceeding of the Fifth International Symposium on Space Mission Operations and Ground Data System: SpaceOps 98, Tokyo, Japan.

[2]Cameron, G. and Marshall, M., 1998, "Exploring the Practical Limits of Operations Autonomy". Proceeding of the Fifth International Symposium on Space Mission Operations and Ground Data System: SpaceOps 98, Tokyo, Japan.

[3]Ferri, P. and Sorensen, E., 1998, "Automated Mission Operations for Rosetta". Proceeding of the Fifth International Symposium on Space Mission Operations and Ground Data System: SpaceOps 98, Tokyo, Japan.

[4]Jones, M., Baldi, A., Melton, B., Bandecchi, M., and Schick M., 1998, "How a Mission Control System was achieved at low cost for a Simple Mission, TEAMSAT". Proceeding of the Fifth International Symposium on Space Mission Operations and Ground Data System: SpaceOps 98, Tokyo, Japan.

[5]Lock, P. and Sarrel, M., 1998, "Distributed Operations for the Cassini/Huygens Mission". Proceeding of the Fifth International Symposium on Space Mission Operations and Ground Data System: SpaceOps 98, Tokyo, Japan.

[6]Monham, A., 1998 "Demonstration of "Lights-Out" Satellite Operations at ESOC". Proceeding of the Fifth International Symposium on Space Mission Operations and Ground Data System: SpaceOps 98, Tokyo, Japan.

[7]Perkins, D., 1998, "NASA Recasts its Operations Approach to Reduce Costs". Proceeding of the Fifth International Symposium on Space Mission Operations and Ground Data System: SpaceOps 98, Tokyo, Japan.

[8]Smith, A.F., 1998, "Multi-Mission Operations Management at ESOC". Proceeding of the Fifth International Symposium on Space Mission Operations and Ground Data System: SpaceOps 98, Tokyo, Japan.

[9]Venkateswarlu, S., Ramalingam, G. and Soma, P., 1998, "Automation of Orbit and Attitude Determination Functions of Indian Remote Sensing Satellite (IRS) Missions". Proceeding of the Fifth International Symposium on Space Mission Operations and Ground Data System: SpaceOps 98, Tokyo, Japan.

[10]Smith, S., 1998, "Automation of Intelsat's Launch Support (LEOP) Operations". Proceeding of the Fifth International Symposium on Space Mission Operations and Ground Data System: SpaceOps 98, Tokyo, Japan.

[11]Truszkowski, W., 1998, "Investigating Human/System Interfaces and Interactions in a "Lights-Out" Operational Environment". Proceeding of the Fifth International Symposium on Space Mission Operations and Ground Data System: SpaceOps 98, Tokyo, Japan.

[12]Walyus, K., Green, S., and Mandl, D., 1998, "Lights-Out Operations for the Transition Region and Corona Explorer (TRACE) Using Operational and Architectural Approaches". Proceeding of the Fifth International Symposium on Space Mission Operations and Ground Data System: SpaceOps 98, Tokyo, Japan.

[13]Fox, J. A., Breed, J., Baker, P., Chu, K., Starr, C., and Baitinger, M., 1998, ´A Web-based Emergency Response Systems for Lights Out Operations´. Proceeding of the Fifth International Symposium on Space Mission Operations and Ground Data Systems: SpaceOps 98, Tokyo, Japan.

[14]Baker, P., Chu, K. Starr, C., Breed, J., Fox J., and Baitinger, B., 1997, "Handling Emergencies in Autonomous Systems with An Episode-Incident-Alert Workflow". Paper presented at the 2nd International Symposium on Reducing the Cost of Spacecraft Ground Systems and Operations, Oxford, England.

[15]Altair Mission Control Systems (Altairis) Web Page. http:// www.altaira.com/html/amcs.html, 1999.

[16]Integrated Test and Operations System (ITOS) Subsystem Capabilities Web Page. http://www510.gsfc.nasa.gov/SMEX/itos_cap/title.htm, 1999.

[17]ASIST Spacecraft Ground System Home Page. http://rs733.gsfc.nasa.gov/ASIST/ASIST-home, 1999.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 332

ISSN 0100-7386
Printed in Brazil

Ground Systems III

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp.333-342
ISSN 0100-7386
Printed in Brazil

Euridis Ground Segment Critical Functions and Main Architecture with a Reusing Approach

Norbert Suard
DTS/MPI/PS/SI - bpi 1501
Centre National d'Etudes Spatiales (CNES)
18, avenue Edouard Belin
31401 Toulouse CEDEX 4, France
suard@cnes.fr

Abstract

*A Ranging service on INMARSAT III AORE, called **EURIDIS,** has been developed by a joint CNES/DGAC program. This development constitutes an in-kind delivery from France to the EGNOS Program, the European Regional augmentation to GPS and GLONASS, based on INMARSAT III AORE and IOR navigation transponders.*
EURIDIS system tests review took place in January 1999.
Before the EURIDIS Program, CNES has developed and conducted an experiment (called CE-GPS experiment) based on the INMARSAT II AORE in order to test some critical functions of a Ranging system.
In view of the positive results of this experiment, some softwares, methods or algorithms used or tested during this experiment are reused by the industrial team for designing the EURIDIS ground system and of course for providing the EURIDIS software.
In order to provide the ground segment software at an attractive cost, the applicative software design is based on COTS products (RTworks, Socks...) or CNES software kit (ZOOM) and the embedded software design is based on TOPSTAR200, a SEXTANT Avionique product.
With the same ideas, an independent measurement station has been developed on CE-GPS stations.
The paper describes briefly the system and gives an overview of the ground system requirements.
Then the reused approach is exposed through several points of Euridis experience (experiment and design, method for critical function identification and valuation of software components, COTS components (hardware and software) within ground segment or software architecture, independent measurement station).
Focus on advantages or drawbacks in reusing COTS software or in-house software will be also presented from EURIDIS experience.
Then the paper concludes that, even with a reusing environment and for a new space-user, civil aviation, new engineering concepts and advanced technology are needed to satisfy requirements.
***Keywords**: Ground Segment, Reusing Approach*

Context

Satellite navigation will be affordable for civil aviation users when their stringent requirements on accuracy, integrity, continuity and availability are provided by GNSS1, a Global Navigation Satellite System based on GPS, GLONASS and Geostationary overlay augmentation service.

EGNOS is the European Regional augmentation to GPS and GLONASS, based on INMARSAT III AORE and IOR navigation transponders. It will provide Advanced Operating Capability in several steps (Ranging, Integrity, Wide Area Differential services) and will be complemented later in order to obtain a Full Operational Capability.

The Ranging service on INMARSAT III AORE has been developed by a joint CNES/DGAC program called EURIDIS. This development constitutes an in-kind delivery from France to the EGNOS Program.

This development, led by THOMSON-CSF/T4S (now ALCATEL) with SEXTANT and SYSECA as co-contractors, and with SRTI and SEMA Group as sub-contractors, started in mid 95 and ended in September 98 with the ground segment test acceptance.

Before the EURIDIS Program, CNES has developed and conducted an experiment based on INMARSAT II AORE in order to test some critical functions of a Ranging system ([1] and [2]).

EURIDIS System

Implementation

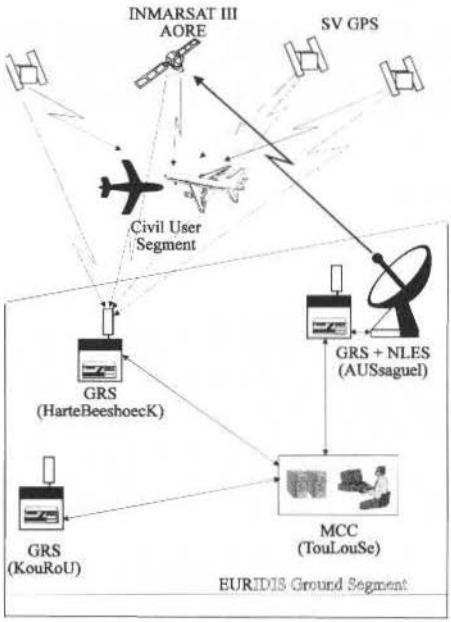


Fig. 1 EURIDIS Segments and Locations

Requirements

The EURIDIS goal is, in the INMARSAT III AORE satellite footprint, to provide through the INMARSAT III L1 navigation transponder, a supplementary Ranging Signal to Civil Aviation users that intend to use GPS as a supplementary navigation means for flight phases from "En Route" up to "Non precision Approach".

The users shall be equipped with RTCA-DO-229 compliant receivers.

To meet this goal, the EURIDIS Signal In Space (SIS) provides accuracy and integrity which better or equivalent to those of a GPS signal with SA, i.e. :

- Signal In Space contribution to user range error (UERE) ≤ 33 m (1σ)
- SIS **critical non integrity < 10^{-5}/h**

Based on the performance capabilities assessed through preliminary studies and experiment [2], the nominal EURIDIS UERE requirement is **UERE ≤ 12 m (1σ)** except during satellite maneuvers.

These performances shall be met everywhere in the EAOR footprint at 5° elevation angle (service zone).

If UERE is higher than 33 m (1σ), EURIDIS Signal In Space will be declared not usable by a "DON'T USE" navigation message.

A critical non integrity (NIC) is considered if the instantaneous UERE, which is mainly composed of orbitography and clocks errors, reaches 100 metres at any place of the service zone without being announced to users in **less than 10 seconds**.

EURIDIS Ground Segment

Design

The EURIDIS ground segment includes :

- Three Geostationary Ranging Stations (GRS) implementing a wide triangular observation base with the stations located in Aussaguel (France), Kourou (French Guyana) and Hartebeeshoeck (South Africa),
- A Mission Control Centre (MCC), located in Toulouse (France), implementing the operating control centre,
- A navigation Land Earth Station (NLES), based on an existing station of the INMARSAT Network, located in Aussaguel and operated by France Telecom, implementing the feeder link to the AORE Navigation payload.
- A ground network to link the stations (GRS and NLES) to the MCC.

Functions

The main functions of EURIDIS ground segment are [6] :

- collection of data needed by the others functions,
- orbitography process including an INMARSAT III satellite dynamic model and a predictive model for assessing the effects of a maneuver during and shortly after such an event,
- relative synchronization of distant ground stations based on the GPS satellites which are in common view of at least two of the three GRS [1],
- synchronization of the GEO time within the GPS constellation time including a GPS time restitution based on all-in-view GPS pseudo-ranges received at Aussaguel and connected to a very precise caesium clock [1],
- management and generation of EURIDIS messages in compliance with SIS specifications [7],
- long loop function including transmission of the signal modulated by a Coarse/Acquisition code and navigation message data, control of the signal code/carrier coherency and the on-board clock transfer,
- monitoring of the EURIDIS integrity,
- monitoring and management of the mission operations.

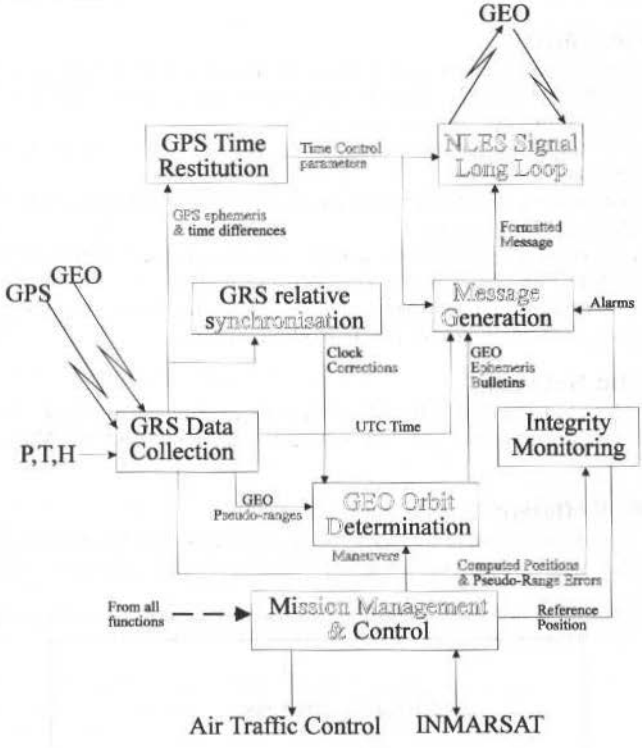


Fig. 2 Functional Diagram

According to the NIC definition (cf. §2), most of the main functions may contribute to a NIC, and this is the reason why, in a first approach, each main function can be classified as a critical one. So, in order to reduce the critical function domain, an analysis, based on reused work (cf. §5.3) has been conducted whose results are presented in §6.3.

Ground Segment Architecture

Architecture of Each GRS

Each GRS includes the same acquisition equipments consisting in :

- 15 channel EURIDIS receivers designed to provide GPS pseudo-ranges, 1 to 3 GEO pseudo-ranges, spectrum analysis data and to use a calibration signal. An omnidirectional antenna

connected to a receiver is used for GPS and a L1 GEO directive antenna is also connected to reduce multipaths and multi GEO interference effects. Each frequency receiver is controlled by an external caesium clock,

- A L1/L2 semi-codeless GPS receiver connected to the omnidirectional antenna for ionospheric correction,
- A meteorological station to deliver pressure, temperature, hygrometry parameters for tropospheric correction,
- A computer to collect and sort data, and to monitor GRS equipements and process.

Architecture of the NLES

The NLES is coupled with the Aussaguel GRS and with the existing C band Inmarsat traffic communication antenna.

In addition to this antenna and to the GRS equipments, the other NLES equipments are :

- The long loop equipment including a GEO signal generator, a calibration signal generator, a specific receiver/loop control equipment, ASIC, a frequency synthesizer and a micro-processor,
- A spectrum analyzer,
- A specific integrity receiver.

Architecture of the MCC

The MCC is composed of five computers linked through a double Ethernet LAN and a hub for communications with GRS and external links (Inmarsat Navigation Center, ATC...).

On each computer, specific functions are running :

- The first one for critical real-time functions (integrity data collection, integrity monitoring, message generation),
- The second one for GEO orbit determination and for time synchronizations (GPS time restitution and GRS relative synchronization),
- The third one for other data collections and for real-time mission management and control,
- The fourth one for batch process (archiving, off-line analysis...)
- The fifth for external communications with secure data proxy data channels.

The MCC contains a sixth computer for the network monitoring.

Architecture of the Network

The GRS and the NLES are linked to the MCC through a redundant and dedicated IP ground network based on the CNES private network. The link between Aussaguel and the MCC is the most critical because of the NLES.

Specific EURIDIS Software

Each station of this ground segment includes specific softwares for EURIDIS, either with a computer (applicative software) or with microprocessor in receivers (embedded software) and the MCC is made of applicative software.

The applicative software is designed according to a multi-layer model (cf. Fig. 3).

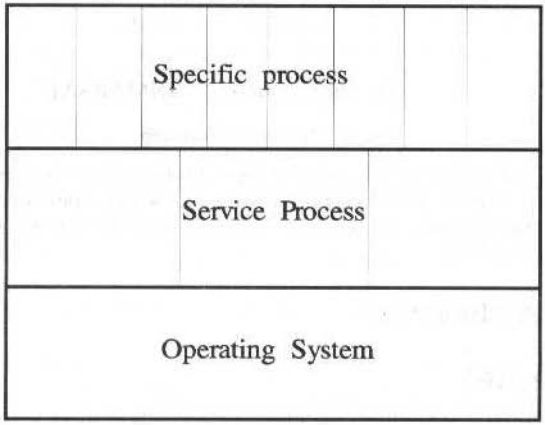


Fig. 3 Applicative Software Model

Reused Components

In order to provide the ground segment and system test acceptance facilities at an attractive cost, the reusing method was applied at several levels :

- definition and design,
- critical function identification,
- equipment choices on the shelf if existing,
- COTS or CNES softwares.

Reuse During Definition Phase

In view of the positive results of the CE-GPS experiment[2], some methods or algorithms used or tested during this experiment are reused by the industrial team for designing the EURIDIS ground system and of course for providing the EURIDIS software. The major function elements reused are :

- broadcasting of a GPS-like signal through a geostationary transponder,
- generation and control of a virtual ultra-stable clock on board the satellite,
- orbitography and orbit prediction of the geostationary satellite with accuracy equivalent to that of GPS (10 metres),
- relative synchronization of distant ground stations,
- GPS time restitution.

Reuse During Design Phase

Strategy

The industrial team has based the ground segment design on existing products.

For equipments, only receivers and long loop equipments did not exist. In these cases, the proposal was based on SEXTANT Avionique products.

The same reuse strategy was applied for the specific EURIDIS software.

The embedded software is based on a SEXTANT Avionique product, the TOPSTAR200 and the applicative software is based on COTS products (RTworks [5], SOCKS_V5 [8]...), a CNES flight dynamics software (ZOOM), a CNES external link manager (PCS).

Some Details on Some Reused Software Facilities

TOPSTAR200 is a user GPS receiver comprising 15 channels including embedded software in accordance with several ARINC or GPS standards.

Its software is modified to obtain the EURIDIS receiver software and the Integrity receiver software. Some basic functions of this software are reused to obtain the long loop equipment software.

RTworks is a family of software products for building client/server applications that manage intelligently time-critical data.

The different modules of RTworks are used separately or together to build systems that acquire, analyse, display and act upon large amounts of changing data. Typical applications of RTworks include network management, alarm correlation, command and control, Human-computer display...

SOCKS is a mechanism by which a secure proxy data channel can be established between two computers.

It is intended for client/server environment. From the client's perspective, SOCKS is transparent. From the server's perspective, SOCKS is a client.

SOCKS consists of two components. The SOCKS server is implemented at the application layer while the SOCKS client is sandwiched between client's application layer and transport layer.

ZOOM is a set of tools for precise orbit determination.

It performs a numerical integration of the satellite motion equations and of the variational equations in a given dynamic context using a high order Cowell fixed step integrator.

For measurements during orbit calculations, ZOOM can take into account several corrections (troposphere, ionosphere, offset of the satellite's mass centre, effects of the antenna phase centre, relativity, Earth orientation)

Parameters for dynamics or measurement models are selected by a user and they are estimated by ZOOM with a least square method.

PCS [12] is a manager of external data links providing security controls in order to allow only necessary data flows between Euridis network and other networks (CNES Intranet, Internet (mail, ftp, www...)).

Reuse During Critical Function Identification

NIC Valuation and Global Strategy

A non integrity (NIC) is considered if the instantaneous User Equivalent Range Error, mainly composed of orbitography and clocks errors, reaches 100 metres in any place of the service zone without being announced to users in **less than 10 seconds** (time-to-alarm). The probability of such a non integrity shall be **less than 10^{-5}/h**.

These quantitative targets are applicable to both hardware and software needed to perform EURIDIS system functions.

For hardware, classical RAMS tools are used to demonstrate that requirements are met.

For the software, no normalized methodology was available to demonstrate compliance of quantitative targets. Consequently, work, standards or rules used by civil aviation corporation were the basis to tackle this problem and defined the EURIDIS strategy.

Some Details on Reused Standards

The JAA's (Joint Aviation Authority) rules defined in the JAR (Joint Airworthiness Requirements) associate potential consequences of one event with a quantified level of occurrence probability :

Consequence	Probability
catastrophic	$\leq 10^{-9}$ per flight hour
dangerous	$\leq 10^{-7}$ per flight hour
major	$\leq 10^{-5}$ per flight hour
minor	$\leq 10^{-3}$ per flight hour

The DO 178-B [3], not applicable to EURIDIS Project, associates potential consequences of software failures with a particular level of development :

Consequence	Development level
catastrophic	level A
dangerous	level B
major	level C
minor	level D

The MPM-53-00-07 CNES specification [11] on « Software Dependability study using the Fault Tree method », a classical analysis method based on a deductive process, enables:

- identification and graphical representation with logical operators (mainly OR gates and AND gates) of the different event combinations (hardware and/or software) that lead to the top, see Fig. 4

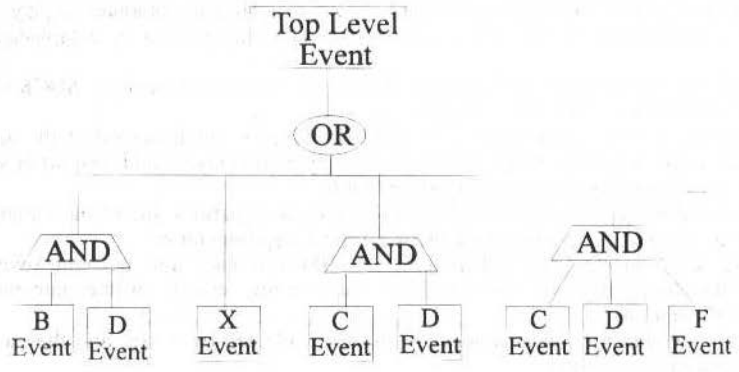


Fig. 4 a fault tree example

- to every event or event combination, allocation of a quantitative target consistent with the target specified for the fear event, see Fig. 6.

Reuse for System Test Facilities

To perform EURIDIS system tests, an independent and transportable station dedicated to data collection and measurements in different places of the INMARSAT III AORE footprint has been provided.

This station provides GPS time restitution linked with an external or an internal clock according to its use (Determination of accuracy of GPS time restitution or Determination of EURIDIS UERE budget for static users).

During the CE-GPS experiment, three stations including some of these functions (and with other functions) were developed. So the EURIDIS measurement station for system test facilities has been based on an upgrade CE-GPS station, realised by the initial developers, IN-SNEC to take benefit of their experience.

EURIDIS Experience

Applicative Software Architecture

The internal architecture of the EURIDIS applicative software is slightly different from the layer model (cf. Fig. 3) due to the real functions and the real implementation of the software reused.

The following figure sums up the EURIDIS applicative software architecture :

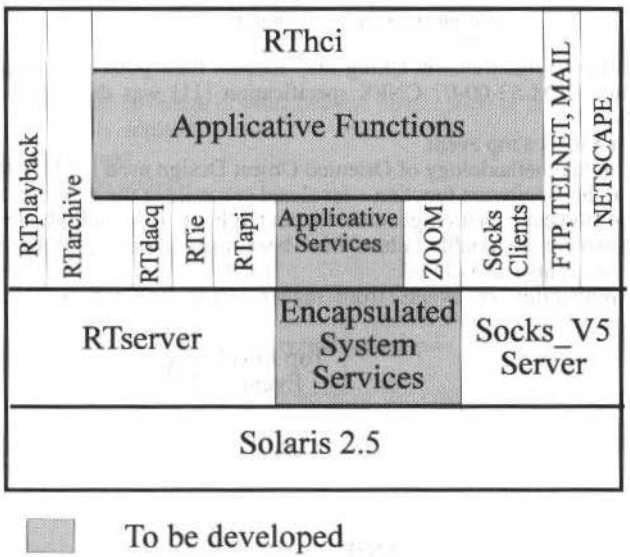


Fig. 5 Applicative Software Architecture

To realize an applicative function, applicative services, RTapi, or SOCKS Clients library are mixed with its own code.

Reusability : a Good Thing for Software?

Reusability is one of the ways to achieve the objective of cost reduction for space mission operations and ground data systems.

CNES led many experiences in this domain of reusability, either for tools, but also today for develop processing centres ([9] and[10]).

For EURIDIS, CE-GPS experiment documents were given and ZOOM reusability was set for the CNES call for tenders and the industrial team has reinforced this reusing approach with the same criteria.

The main advantages of this reusing approach are the following :

- development cost reduction
- time delivery reduction
- reduction of development risks.

On the other hand, this approach is only credible when the development team has at its disposal :

- a specific training on the reused tools or a specialized support (a reused tool développer) ,

- an efficient technical assistance given by the COST or in-house tool provider,
- enough time to design the software architecture in order to mix reused software and new software,
- a need attribute (essential, negotiable) for each user requirement in order to determine according to a cost analysis if a need which is not covered by the tools chosen, has to be developed through a new software or just through a tool modification or has not be developed (negotiable need),
- tools based on standards or code-ownership to ensure the durability of the applicative software (several years).

NIC Valuation

Using the two civil aviation approaches (cf. 5.3.2.), a strategy has been defined which consists in associating, to every software which can participate in a NIC occurrence scenario, a development level consistent with the occurrence probability target :

Probability	Development level
$\leq 10^{-9}$ per flight hour	level 1
$\leq 10^{-7}$ per flight hour	level 2
$\leq 10^{-5}$ per flight hour	level 3
$\leq 10^{-3}$ per flight hour	level 4

At this step, a software classification taking into account their potential impact on NIC occurrence was necessary and the MPM-53-00-07 CNES specification [11] was the basis of this work with the following consideration :

→ occurrence of a NIC is a top event.

In relationship with the methodology of Oriented Object Design used for EURIDIS, the fault tree has been first developed at the « software function » level and secondly at the « software operation » level.

After reducing the Boolean equation generated from the Fault Tree, only the level one cut set (single failure) and the level two cut set (double failure) have been taken into account. In the double failure case, the independence of the events was verified.

The occurrence probability of events from higher levels (triple failure ...) was considered as negligible.

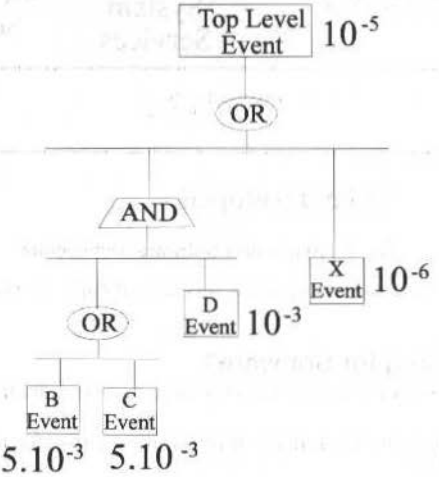


Fig. 6 Example of a reduced fault tree with quantitative targets

This analysis has demonstrated that the quantitative targets allocated to the EURIDIS Software have to be between 10^{-5}/h and 10^{-3}/h.

In accordance to the methodology chosen, the development of level 3 and 4 has been specified for the EURIDIS Software.

The software or software operation which has to be developed at the third level was called later on « critical software » or « critical operation » .

Due to the project context, a common frame of development requirements has been introduced for the whole software [4]. As the development of level 3 is intended to be more secure, their requirements contain more precise points.

The main characteristics of those levels are given hereafter.

Level 4 :

- Key points all along the development life cycle
- Traceability matrix between each phase from requirements to coding (down phases) and from coding to testing (up phases) in relationship with the matrix of the same level during the down phases
- Rules and advises at each step from requirements to tests
- Configuration management for all software elements and associated documents
- HCI demonstration models during conception
- Computer means and performance margins evaluation as soon as possible and updating
- Formalised quality activities for each step including controls, readings, inspections, tests, tools usage
- Formalised management of modifications, waivers, anomalies, and every activities productions.

Level 3 :

Same as level 4 plus :

- unit testing with maximal coverage of branches and justification of those not covered
- coding rules more strictly applied
- quality activities reinforced
- double readings of all the codes.

Conclusion

Within a cost reduction environment, new engineering concepts and advanced technology are always needed to satisfy requirements. For EURIDIS, the reusing approach for a new system (a GPS like ranging system) or for a criticity valuation method (NIC valuation) shows that reusing is compatible with new solutions.

The reusing approach is an attractive way to achieve the cost reduction objective. The EURIDIS project team has planned this approach since the beginning of the industrial work packages. The conditions necessary for the success of this approach have not been forgotten and have been applied.

Acronyms

AORE :	Atlantic Ocean Region - East
CNES :	Centre National d'Etudes Spatiales
COTS :	Commercial On The Shelf
DGAC :	Direction Générale de l'Aviation Civile - French Civil Aviation
EGNOS :	European Geostationary Navigation Overlay System
GEO :	INMARSAT III AORE satellite
GLONASS :	GLObal NAvigation Satellite System (Russian system)
GNSS :	Global Navigation Satellite System (Generic term)
GPS :	Global Positioning System
GRS :	Geostationary Ranging Station
HCI :	Human Computer Interface
IOR :	Indian Ocean Region
MCC :	Mission Control Centre
NIC :	Critical Non Integrity (french acronym)
NLES :	Navigation Land Earth Station
RAMS :	Reliability, Availability, Maintenability and Safety
SIS :	Signal In Space
UERE :	User Equivalent Range Error

References

[1] European complement to GPS : presentation of the concept and experimental system, J. Barbier and T. Trémas (CNES), 7th European Frequency and Time Forum, Neuchâtel, march 93

[2] European complement to GPS : main experimental results, J. Barbier, M. Deleuze (CNES) and co-authors, session A4, ION GPS-94, september 20-23, 1994

[3] DO-178B / E12B : considérations sur le logiciel en vue de la certification des systèmes et équipements bord, RTCA standard

[4] CT/AQ/QL/RS 94-516 : exigences qualité applicables aux logiciels EURIDIS, J.F. Poble (CNES)

[5] RTworks : Technical Overview, RTworks is a registered trademark of Talarian Corporation

[6] Design and Operations of the EURIDIS System, P. Gouni, D. Flament (Thomson-CSF), H. Secretan, A. Job (CNES), DSNS'96, St Pétesbourg, May 96

[7] RTCA/DO-229 MOPS RTCA, Appendix AandB

[8] What is SOCKS : www.socks.nec.com (HTTP@). SOCKS is a registered trademark of NEC USA, Inc

[9] SPOT Ground Segment Re-engineering Program, P. Lods (CNES), SpaceOps 96, Munich, september 96

[10]Low cost processing centres for two CNES experiments, N. Suard (CNES), Symposium on Reducing the Cost of Spacecraft Ground Systems and Operations, Chilton, september 95

[11]MPM-53-00-07, Analyse de Sûreté de Fonctionnement du logiciel par la méthode de l'arbre des causes,CNES specification

[12]EURIDIS - External links and security aspect, R. Moreno (CNES), DASIA 98, Athens, may 98

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp.343-349

ISSN 0100-7386
Printed in Brazil

An Economical Approach to EGSE Development

Berti Brigitte Meisinger
German Aerospace Centre (DLR)
Muenchner Str. 20
D-82234 Oberpfaffenhofen
berti.meisinger@dlr.de

Michael Gerhard Kloë
CAM GmbH
Rudolf-Diesel Weg 7ª
D-82205 Gilching
michael.kloe@cam-comp.de

Abstract

The Central Check-out System (CCS) of the CHAMP Satellite was developed by the German Space Operations Centre (GSOC) at the DLR (German Aerospace Centre) in Oberpfaffenhofen.
The CCS, together with a number of so-called SCOEs (Special Check-out Equipment), comprise the CHAMP EGSE (Electrical Ground Support Equipment), the purpose of which is to provide a test bench for the CHAMP satellite during pre-flight testing.
The prime functions of the CCS are:

- *SCOE monitoring and control,*
- *Telemetry acquisition, processing, display and distribution,*
- *Telecommand generation and verification,*
- *Archival and retrieval of all relevant data,*
- *Manual or automatic test execution.*

The CCS is fully compatible with the European Space Agency (ESA) packet telemetry and telecommand standards[1,2].
GSOC has extensive experience in manned and unmanned space projects (the Spacelab missions, AMPTE, ROSAT, TV-SAT, etc.). In view of the high degree of similarity between the standard spacecraft ground control software used at GSOC and the required CCS functionality, it was decided to adapt as many of the available ground system modules for use in the CCS as possible, thus saving development costs.
The CCS was delivered on schedule in June 1998 and is now being used for the integration phase, which will last until the end of 1999.
Since GSOC is also responsible for both CHAMP flight operations and provision of the CHAMP ground segment, the CCS software is currently being adapted for use in the CHAMP mission operation system (MOS), thus doubly benefiting from development cost savings.
This paper describes

- *The required functionality of the CHAMP EGSE/CCS,*
- *The existing GSOC ground control software components which were adapted for incorporation in the CCS,*
- *The CCS architecture.*

The reason for publishing this paper is to describe how control centre software can be efficiently applied to satellite system integration and testing.
Keywords: *Satellite, Ground System, Engineering, EGSE, Mission Operations, Esa, Telemetry, Telecommand, Monitoring, Control, TCP/IP.*

Introduction

CHAMP is a scientific satellite, designed to examine the Earth's gravitational and magnetic fields and to perform atmospheric/ionospheric sounding. The CHAMP mission will contribute to the understanding of geophysics, geodesy, oceanography, meteorology, navigation and the global climate change.

Since CHAMP is a low budget project, it must satisfy two important criteria: low cost development in a short time. It was essential to apply a fast and cost-effective method of developing both the CCS and the MOS.

It was the task of GSOC to develop the CCS, a central part of the EGSE, to support the various test phases of satellite development. This was the first time that DLR/GSOC had been asked to provide not only a ground operations facility but also a major part of the EGSE.

Due to the many years of ground control system development and mission operations experience at GSOC, a complete set of mission operations software was available which could be adapted for use within the CCS. In addition, it was important for the MOS (also under GSOC responsibility) to inherit as much as possible from the CCS software and databases.

Overview of the CHAMP EGSE

The CHAMP EGSE, designed to support satellite electrical integration, system level functional testing and launch preparation, consists of:

1 x CCS (Central Check-out System)
1 x TT&C (Telemetry Tracking and Command) SCOE
1 x AOCS (Attitude & Orbit Control System) SCOE
1 x Power SCOE
6 x Payload or Instrument SCOEs.

The SCOEs are not subject of this paper; they were developed independently of the CCS by each of the spacecraft subsystem and payload instrument suppliers.

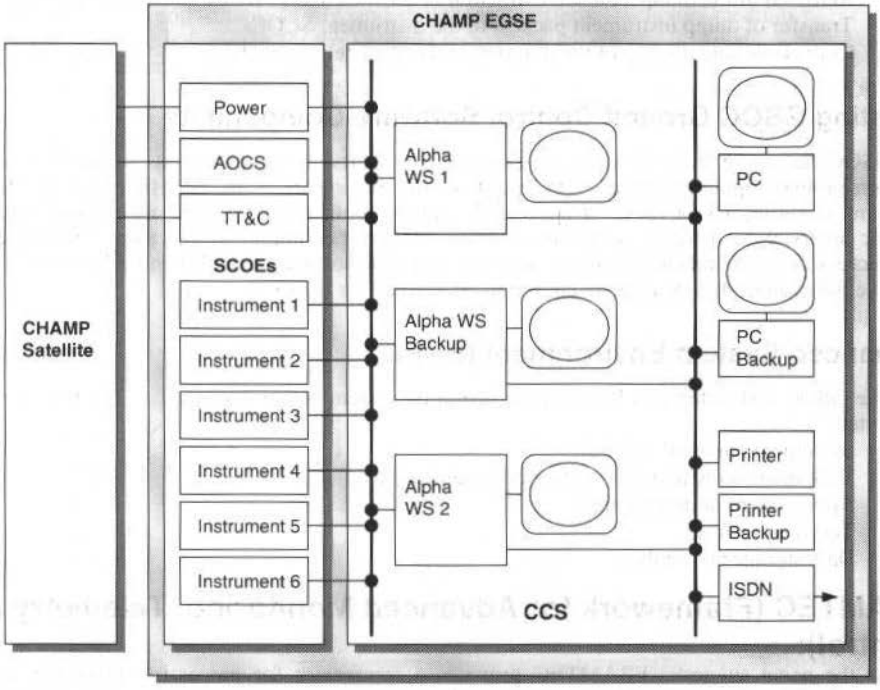


Fig. 1 CHAMP EGSE Overview

An overview of the CHAMP EGSE is shown in figure 1.

The TT&C, AOCS and Power SCOEs can operate either in stand-alone mode or under CCS control. In the latter case, SCOE status data are sent in the form of monitor blocks to the CCS for display; equally, control blocks can be generated on the CCS and sent to a specified SCOE to change its status or initiate a SCOE activity.

The TT&C SCOE, in addition, acts as the interface between the CCS and the satellite during ground operations. Telemetry data are sent from the satellite via the TT&C SCOE to the CCS and telecommands are generated within the CCS and transferred via the TT&C SCOE to the satellite. CHAMP telemetry and telecommand data are in accordance with the ESA standards. The telemetry transmission rate is 1 Mbps, of which 32 kbits are real-time data and the rest are satellite dump data (previously stored on-board).

The CCS can be used in three different modes when testing the CHAMP satellite.

The real-time mode, providing:

- CHAMP telemetry processing:
  - Acquisition and storage of the 1 Mbps teemetry data stream,
  - Extraction of the 32 kbits real-time telemetry (virtual channel (VC) demultiplexing),
  - Reconstruction of the CHAMP telemetry packets,
  - Conversion and display of the housekeeping telemetry,
  - Transfer of the instrument packets to the instrument SCOEs,

- Generation and transmission of telecommands to the satellite,
- Monitoring and control (M&C) of the TT&C, AOCS and Power SCOEs,
- Logging of all data exchanged,
- Manual or automatic test execution.

The playback mode, providing:

- Retrieval of data logged from previous tests (monitoring and control blocks, telemetry packets),
- Display of logged housekeeping telemetry,
- Transmission of logged instrument packets to the instrument SCOEs.

The dump mode, providing:

- Retrieval of the dump data and extraction of the different CHAMP telemetry packets,
- Transfer of dump instrument packets to the instrument SCOEs,
- Conversion and display of the dump housekeeping telemetry data.

Existing GSOC Ground Control Software Components

GSOC has more than 30 years of experience in software development for satellite and manned mission ground support systems. The result of this experience is an extensive suite of flight proven software, covering a vast range of spacecraft support applications. GSOC maintained a modular and generic approach to software development from the very beginning, so that today's software engineers have access to a comprehensive library of easily adaptable software modules and subsystems.

The following subsystems were chosen to be adapted for the CCS.

Enhanced System Environment (ESE):

The GSOC ESE interfaces between the operating system and an application. The following tasks are supported:

- System and network control,
- Task monitoring and task-to-task communications,
- System internal data routing,
- Test diagnostics,
- Data storage and replay.

FRAMTEC (Framework for Advanced Monitoring, Telemetry and Control):

As the name suggests, FRAMTEC provides a framework for processing incoming data and for generation and transmission of control data. Processing and control information is retrieved from a database configured for the application.

The main features of FRAMTEC are:

- Data stream processing, with validation, decommutation, conversion, limit control, etc.,
- Generation and transmission of control data,
- Database configuration (ACCESS-based editor).

TIGRIS (MMI - System):

This GSOC MMI (Man Machine Interface) application supports the presentation and display of application data and the transfer of operator inputs to application tasks. A series of images/data can be designed as view hierarchies. Data are displayed in textual or graphical form in conformance with modern user interface standards (OSF/MOTIF).

Core Command System:

The Core Command System provides the following functions:

- Telecommand information stored in a configurable database,
- Single or sequence telecommand generation and transmission in accordance with ESA standards,
- Telecommand validation and verification.

CCS Hardware Architecture

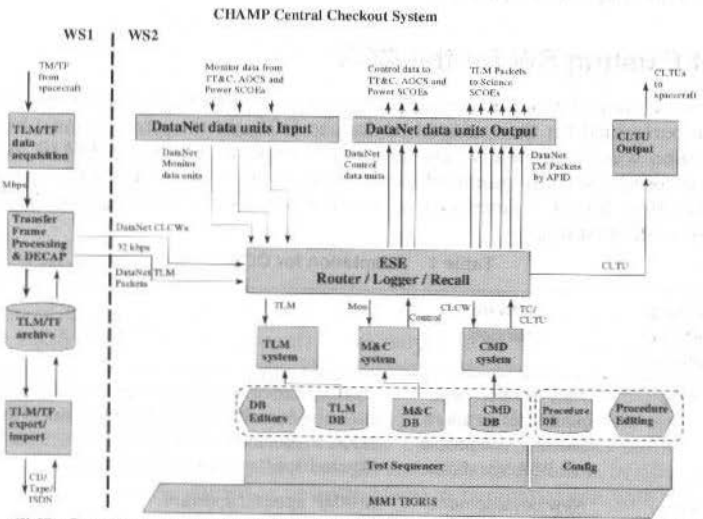
The CHAMP CCS hardware comprises (see figure 1):

- 3 Digital Alpha Workstations (WS), each with 21" monitor,
  - 1 WS for acquisition and archival of the 1Mbps telemetry and extraction of the real-time stream,
  - 1 WS for housekeeping data monitoring, SCOE monitoring and control, telecommand generation and test sequencing,
  - 1 back-up WS, capable of replacing each of the other two WS,
- 2 display stations (PCs), of which 1 is a back-up,
- 2 network laser printers, of which 1 is a back-up,
- 1 ISDN converter, to permit remote access,
- 1 DAT tape drive, connectable to any workstation, as an external archiving medium.

The back-up hardware is foreseen for the required tests at the launch site in Russia, where hardware maintenance or procurement of replacements will be problematic.

Two LANs are provided – a SCOE LAN and a display LAN. All components within the CCS and also the SCOEs are connected via Ethernet. TCP/IP (stream sockets) are used for communications between the CCS and the SCOEs.

Software Architecture



Abbreviations: CMD=Command, TLM=Telemetry, TF=Transfer Frame, CLTU=Command Link Transmission Unit, CLCW=Command Link Control Word, DB=Database.

Fig. 2 CCS Software Architecture

CCS Software Elements:

The processes shown in the left column of figure 2 run on WS 1. These processes are performed by TFPROC and DECAP.

TFPROC (Transfer Frame Processing) performs 'level 0 processing' and archival of telemetry data. As input, complete transfer frames are received from the TT&C SCOE. TFPROC checks the frame validity, archives each frame, demultiplexes the virtual channels and extracts the CLCW (Command Link Control Words).

DECAP extracts the CHAMP telemetry packets from the output of TFPROC. The DECAP software package is required because of the special format of the CHAMP data.

The remaining processes shown in figure 2 reside on WS2.

The ESE is used within the CCS for task-to-task and network-wide communication as well as for logging of all data. It can also be used to recall all logged data into the system to reproduce test results without the attached satellite (cf. 'playback mode'). ESE offers a number of features relevant for diagnostics.

The TLM (Telemetry) System receives CHAMP application packets containing housekeeping telemetry. The system processes all telemetry parameters as defined in the TLM process database and transfers all relevant data to the MMI TIGRIS. The TLM system is based on the FRAMTEC software.

The M&C (Monitor and Control) System enables remote control of the SCOEs from the CCS for coordinated SCOE operation. Monitor blocks are received from, and control blocks sent to, the TT&C, AOCS and Power SCOEs, according to parameters defined in the M&C process database. The M&C system is also based on the FRAMTEC software.

TIGRIS is used within the CCS for display of TLM and M&C data using current day presentation techniques.

The CMD (Command) System generates ESA standard commands (CLTUs - Command Link Transmission Units) for the CHAMP satellite. After command transmission, the CMD system processes CLCWs (Command Link Control Words) from the TFPROC software and performs the COP (the ESA standard Command Operation Procedure) to verify that the commands have reached the satellite correctly. The CMD System is based on the GSOC Core Command System software.

The Test Sequencer permits execution of prepared tests. Test procedures for the different satellite subsystems can run in sequences, linked to time or logical events. The procedures are written in a test control language, providing access to parameter values (TLM and monitoring) and telecommand/control block generation; execution structures for loop or condition handling can be used.

The M&C, telemetry and telecommand databases were configured by the SCOE and satellite manufacturers after GSOC training. It was their responsibility to prepare the parameter and telecommand definitions and data unit structure information.

Adaptation of Existing SW for the CCS

Table 1 shows which new developments were necessary and which modifications to existing software packages had to be performed for their adaptation for the CCS. At this point it must be stated that only the DECAP application was a completely new development specifically for CHAMP. All other software components could be used with only minor adaptations or, in the case of TFPROC, can be used in future missions. The main effort for CCS development was the population of the databases for configuration purposes and test procedure editing.

Table 1 Adaptation for CCS

Software Module	Existing Software Package	Software Changes	Remarks
TFPROC	None	New development; generic for all future missions; compatible with ESA standards.	This application was developed for both CCS and MOS at the same time. Only the different output of the CHAMP TT&C SCOE in relation to the output of the GSOC front-end required special interface handling.
DECAP	None	New development.	CHAMP specific software.
TLM System	FRAMTEC	None.	No software was changed, the configuration was defined in the TLM database.
M&C System	FRAMTEC	None.	No software was changed, the configuration was defined in the M&C database.
MMI System	TIGRIS	None.	Display database had to be generated.
CMD System	Core Telecomm and System	CHAMP specific command structure.	Telecommand database was provided.
Test Sequencer	Operations Scheduler	Adaptation to Open VMS and new interfaces.	This sequencer was originally developed for the automation of operations at the Usingen Control Centre of German Telekom.
Router, Logger, Recall	ESE	Adaptation of TCP/IP IO-Handler	Configuration Tables were updated.

Table 2 Adaptation for MOS

Software Module	CCS Heritage	Reduction of Software Development Effort	Reduction of Configuration Effort	Remarks
TFPROC	Yes	75%	N/A	Adaptation to GSOC front-end.
DECAP	Yes	95%	N/A	Adaptation to GSOC multi-mission environment.
TLM System	Yes	No software changes necessary.	85%	Flight specific parameters to input into database.
MMI	Yes	No software changes necessary.	80%	Additional displays for flight necessary.
CMD System	Yes	No software changes necessary.	95%	Adaptation to GSOC multi-mission environment.
ESE	Yes	No software changes necessary.	95%	Adaptation to GSOC multi-mission environment.

Use of CCS Software in the CHAMP Mission Operations System

CHAMP mission operations will be performed at GSOC and the latter is responsible for developing the CHAMP Mission Operations System (MOS), the ground control system for mission operations.

Due to the generic nature of the software developed by GSOC, it will be possible to use a subset of the CCS in the MOS. Table 2 shows the resulting savings expected. (As only MOS relevant components are listed, neither the M&C system nor the Test Sequencer Software is included.)

As can be seen, almost all the CCS software and databases can be used within the MOS. Only minor changes are necessary to adapt the CCS software to the GSOC multi-mission environment and to finalize the CHAMP databases with flight specific elements.

Economical Advantages

The derivation of the CCS from GSOC ground control software components, and the subsequent derivation of the MOS from CCS components, obviously lead to substantial savings in time and cost when compared with the resources which would have been necessary for an independent development of the two systems.

A further significant factor in cost reduction is attained in the field of satellite operator training. In the CHAMP project, the EGSE/CCS operators will become the satellite operators during the mission. These personnel will work with the CCS during the 9 month assembly, integration, verification and test phase. Important experience will thus be gained at an early stage in familiarisation with the CCS, the satellite and its interfaces. When the same system is then used during actual satellite operations, many hours of training will have been saved and the CCS experience will have imparted a high degree of confidence to the operators in the area of CHAMP satellite control.

Conclusion

The CHAMP project was the first time for GSOC that the classical approach of separating EGSE and ground control system development was not enforced.

GSOC is responsible for providing both a CCS (for the CHAMP EGSE) and a CHAMP ground control system (MOS), following a strict design-to-cost approach.

The method applied was to firstly, adapt already existing GSOC software for incorporation in the CCS and secondly, adapt the CCS software for incorporation in the MOS.

This approach provides the following main advantages:

- Substantial cost savings in relation to independent development of a CCS and MOS by different suppliers,
- Simplification and reduction of interfaces in the project, since the CCS and MOS are based on the same system,
- Prime components of the MOS are already tested during EGSE/CCS operations in the satellite test phase,

- Satellite operators can gain familiarity with satellite behaviour and control using mission software during the ground test phase.

The benefits gained in having a single supplier for both EGSE and mission operations systems should be further exploited.

Acknowledgements

We would like to thank Dr. P. Piotrowski, N. Jansen, Dr. C. Tilgner and C. Ward for their help and support in the compilation of this paper. The project management of the CHAMP mission is under the responsibility of GFZ (GeoForschungsZentrum Potsdam).

References

[1] ESA Packet Telemetry Standard, ESA PSS-04-106, January 1988
[2] ESA Packet Telecommand Standard, ESA PSS-04-107, April 1992

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp.350-357

ISSN 0100-7386
Printed in Brazil

The ENVISAT Payload Data Segment

Michel Irle
Alcatel Space Industries
283 Rue de la Minière
78533 Buc France
michel.irle@T4S.thomson-csf.fr

Jean Pierre Guignard
(ESA/Esrin)
Via Galileo Galilei
00044 Frascati Italy
jean-pierre.guignard@mail.esrin.esa.it

Abstract

ENVISAT is the major Earth Observation Program managed by the European Space Agency. Within this program Alcatel Space Industries is Prime Contractor responsible for the development of the Payload Data Segment (PDS) which provides all services needed for the exploitation of the data produced by the ENVISAT Instruments.
After a short description of the ENVISAT mission this paper describes the system design approach used by Alcatel for designing a highly configurable and scalable PDS architecture.

Envisat-1 Mission

Basic Mission Objectives

The main objective of the ENVISAT-1 programme is to provide Europe with an enhanced capability for remote sensing observation of the Earth from space and to increase the opportunity for participating states to take part in the monitoring and studying of the Earth and its environment.

Overall Operational Mission Objectives

The primary operational mission objectives of ENVISAT-1 are to :

- Provide for continuity of the observations started with the European Remote Sensing Satellites (ERS-1 and ERS-2) for monitoring of coastal zones, open oceans, ice, and land surface processes including those obtained from radar-based observations.
- Provide for enhancement of the ERS-1 mission with observation of biological components and colour of the oceans.
- Extend the range of parameters observed to meet the need to increase knowledge of the factors determining the environment.
- Make a significant contribution to environmental studies, notably the areas of atmospheric chemistry and ocean studies including marine biology.

In addition, there are two related, but secondary, mission objectives:

Enabling more effective monitoring of the Earth's resources as a means for their improved management; and

- Providing an expanded understanding of solid Earth processes.

In addition to continuing and improving upon measurements initiated by ERS-1 and ERS-2, the ENVISAT-1 mission will take into account the requirements related to the global study and monitoring of the Earth and its environment as expressed by international co-operative programmes such as the International Geosphere and Biosphere Programme and the World Climate Research Programme. The ENVISAT-1 is an essential element in providing long-term continuous data sets that are essential for addressing environmental, climatological, and geological issues.

Instruments

In order to fulfil it's mission, the ENVISAT-1 payload will be composed of the following instruments

- ESA Developed Instruments (EDIs) :
  - The Advanced Synthetic Aperture Radar (ASAR) will make all-weather observations of surface characteristics.
  - The Global Ozone Monitoring by Occultation of Stars (GOMOS) will measure trace-gas concentrations.

- The Laser Retro-Reflector (LRR) will allow ground-based laser stations to measure the range of the spacecraft precisely.
- The Medium-Resolution Imaging Spectrometer (MERIS) will allow to measure 'ocean color'.
- The Michelson Interferometer for Passive Atmospheric Sounding (MIPAS) will allow to observe the atmosphere between tangential heights of 5 and 150 km.
- The Microwave Radiometer (MWR) will provide data required to correct the data from the Radar Altimeter (RA-2).
- The Radar Altimeter (RA-2) will make precise measurements of altitude over the entire Earth, including all types of surfaces.

- Announcement of Opportunity Instruments (AOIs):
  - The Scanning Imaging Absorption Spectrometer for Atmospheric Cartography (SCIAMACHY) will allow to measure the abundance of atmospheric constituents in both the troposphere and the stratosphere.
  - The Advanced Along-Track Scanning Radiometer (AATSR) will measure global sea-surface temperatures.
  - The Doppler Orbitography and Radio-positioning Integrated by Satellite (DORIS) will allow to determine the satellite exact position in space.

Ground Segment

There will be two major elements within the ENVISAT-1 ground segment :

- The Flight Operation Segment (FOS): it will be responsible for spacecraft command and control.
- The Payload Data Segment (PDS): it will be responsible for overall mission planning and for the acquisition, processing, archiving and dissemination of instrument data, as well as for the provision of a user's service to provide product and data access services.

Centres / Stations Definitions And Locations

The PDS comprises the following Centres / Stations (C/S) :

- The Payload Data Control Centre (PDCC) at ESRIN, responsible for the instruments and ground segment planning and for the overall PDS monitoring and control. It also co-ordinates user services and provides quality and engineering support for the products.
- The Payload Data Handling Stations (PDHS) at ESRIN and Kiruna acquire measurement data downloaded by the spacecraft (respectively in Ka and X-band), process it and disseminate products, according to PDCC directives. A short term rolling archive is provided. Local services are also offered to users.
- The Payload Data Acquisition Station (PDAS) at Fucino acquires only measurement data (in X-band) and forwards it to the PDHS-ESRIN station for processing.
- The Low rate Reference Archive Centre (LRAC) at Kiruna archives and processes the global low rate data off-line, to provide a complete consolidated Low-Rate level-1b data set for EDIs (ESA Developed Instruments) and ASAR-LR Instruments, and a pre-consolidated level-1b for other AOs (Announcement of Opportunity) Instruments (level-1b are geolocated engineering calibrated products). Local services are also offered to internal users.
- The Processing and Archiving Centres (PACs) located at ESA member states, archive and process off-line High-Rate data and generate off-line geophysical products for regional High-Rate or Full Resolution instruments and global Low-Rate instruments. There are 6 PACs : UK-PAC, D-PAC, I-PAC, F-PAC, S-PAC and E-PAC. The Finnish Meteorological Institute (FMI) is associated with the D-PAC.

- The National Stations, located in ESA member states, acquiring data and providing ESA services (NS-Es).

Figure 1 shows where the PDS Centres and Stations are. The main functions are listed for each Centre or Station, except for PACs, where instruments are listed.

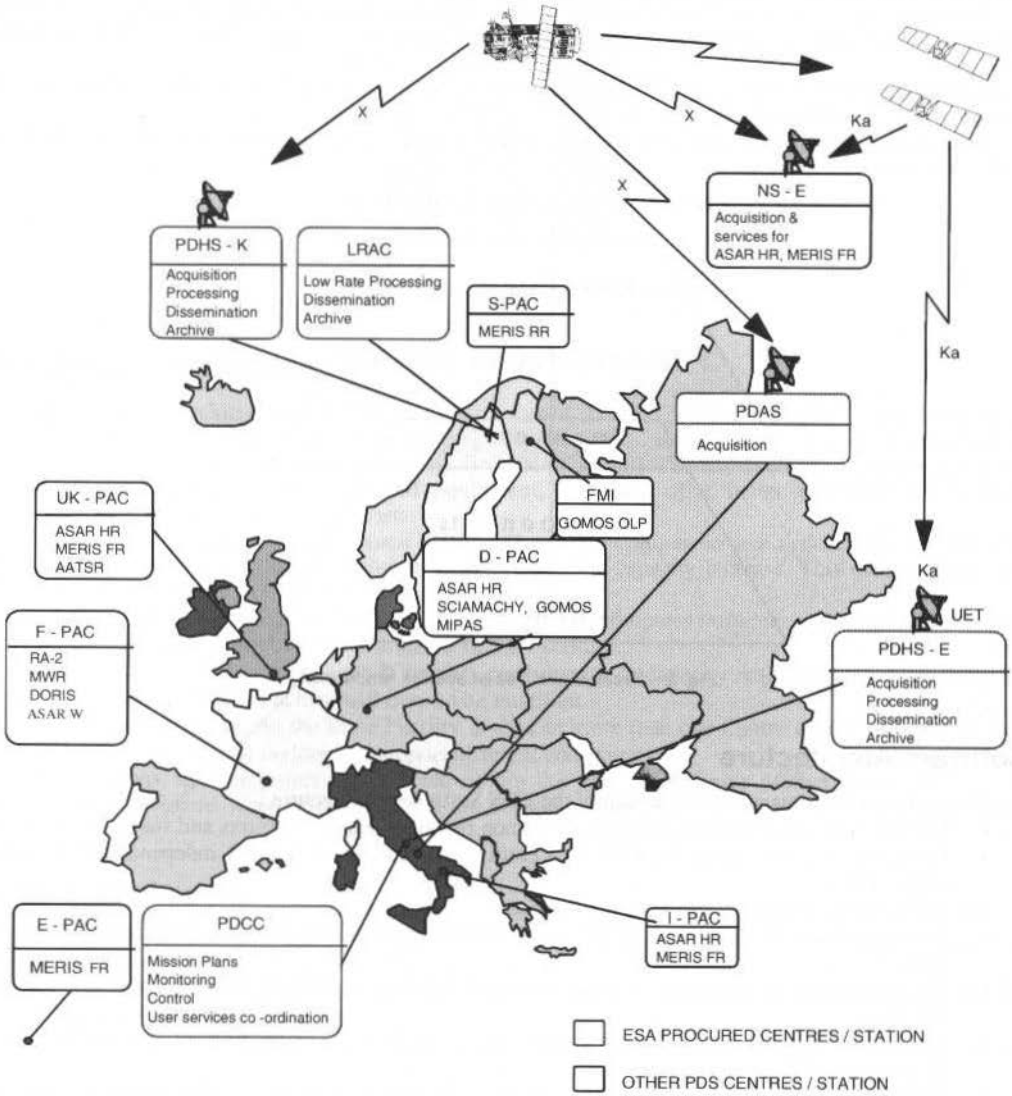


Fig. 1 PDS Centres/Stations locations with their associated functions

System Design Concepts

Constraints on System Design

When designing and developing the PDS system, the design and development of similar functions that exist in different geographical locations must be avoided.

A function may exist in different centres and stations, for example :

- Users services are offered by all centres and stations.
- Low bit rate processing is performed, in similar ways in the two PDHSs and in the LRAC.
- ASAR processing is performed in the two PDHSs and in three PACs (UK, D and I PAC).

Some functions must also be available as Generic Elements to be integrated in the PACs and NS-E, in order to provide the same services to users, independently of the centre or station that provides it.

To satisfy these objectives, the PDS architecture is broken down in development components called Facilities, which will be used to build the PDS centres and stations, and will be available for PAC assembly.

Centres/stations are co-located sets of PDS Facilities. The generic elements are the Facilities, which are common to the PDS and external systems such as PACs.

From a top-down approach the PDS is considered as a set of centres and stations, each constituted by a set of Facilities :

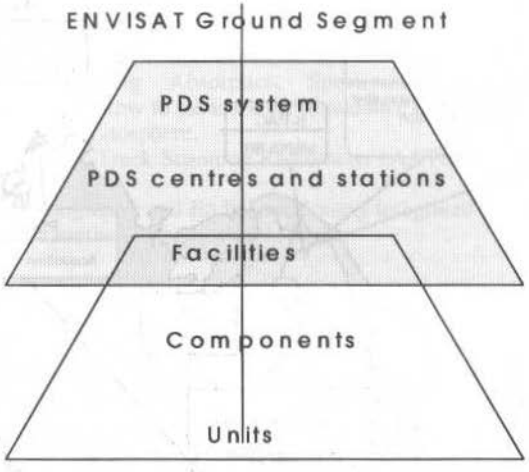


Fig. 2 Break down tree of system functions.

Software Architecture

Two major constraints affect the design of the PDS Software architecture :

- The need to have generic elements and common functions between centres and stations.
- The need to break down the PDS into entities that can be developed independently by sub-contractors.

The main drivers for the definition of facilities are :

- The regrouping of functions in order to obtain facilities that perform specific functions or services.
- The regrouping of functions in order to avoid function overlapping or split between several facilities.
- The need to have an homogeneity in the sizes of facilities, in order to have roughly the same development effort.
- Orientation of facilities towards one aspect of technology (such as, Monitoring & Control, Archiving, user's interface, etc.).

Processing Architecture

The system design shall also take into account the major characteristic of ENVISAT : to process data from 7 different instruments.

The processing algorithms are gathered on an instrument-by-instrument basis, in processing chains for ASAR, MERIS and each Low Bit Rate (LBR) instrument.

Grouping of algorithms by instrument allows the development tasks to be optimised, requesting to each development team only the knowledge of one particular instrument. This guarantees homogeneity between the products for a given instrument.

The common functions between the processing chains will only be developed once, to provide the interfaces between the algorithms and the other PDS functions.

The common functions will act as an Host Structure on which the processing chains will be connected.

The Figure 3 presents this concept.

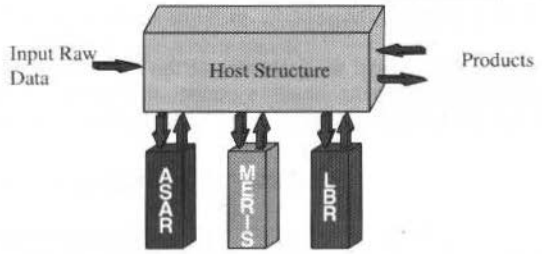


Fig. 3 Processing Function Modular Design

Hardware Architecture

The hardware architecture is as far as possible based on *Off-The-Shelf Computers and peripherals*. The PDS is a large, complex software system, and except for some acquisition and synchronisation functions, all the functions are supported by available Off-The-Shelf (OTS) components. OTS components are cheaper than developed equipment, and they offer better evolution and upgrade capabilities, with lower maintenance costs.

Hardware Architecture design is such that the *Hardware Equipments are dedicated to one Facility*. Hardware costs are going down but the performance and the capacity increase. This design choice will have little impact on the overall cost, but it :

- Eases the integration. The Facilities do not share any computer resources, only the local networks are common resources to all or a subset of the Facilities.
- Gives flexibility to the design. If Facility performance or capacity requirements change, only the architecture of this Facility will have to be modified.
- Offers scalability. As the same Facility is used in more than one Centre or Station, its Hardware architecture can be taylored to support different configurations.

Workstations and computers are selected within the offer of a *Single Manufacturer*. This choice avoids major problems when interconnecting the different equipment, and eases the maintenance during operations.

Operational costs

The system architecture that comes from the system design is efficient with respect to operational costs. The major contributions to the operational costs are :

- Maintenance : one hardware manufacturer is selected, and the equipment chosen is up-to-date, the maintenance costs can be kept low.
- Staffing : the involvement of operators is limited to tasks that require special expertise or decisions, all routine activities are fully automatised.
- Media : the choice of media for archiving and product dissemination between centres is made from performance requirements and cost-reduction goals.

The Facilities (system design entities)

To identify the independent functions of the PDS, it is helpful to look at it from the system point of view, and from the physical implementation in stations and centres point of view.

The system point of view is useful to define the system functions, but does not take into account the stations and centres specific constraints, and the design drivers that have been retained.

The next section explains how the facilities have been defined taking into account design constraints and drivers and using the two points of view of the system.

The two points of view of the PDS

The two points of view of the PDS are shown in figure 4, the columns depict the five main system functions and their main subfunctions, (all the functions are not depicted) the raws depict the four

centres. Where a raw crosses a column, the specific instance of the system function for that centre is listed.

From the figure 4, it can be seen where the instances of the independent functions are placed in the different centres. For example, the archiving function occurs as a rolling archive in a PDHS, an LBR archive in the LRAC and a product archive in a PAC, but basically the same function is performed in all these centres.

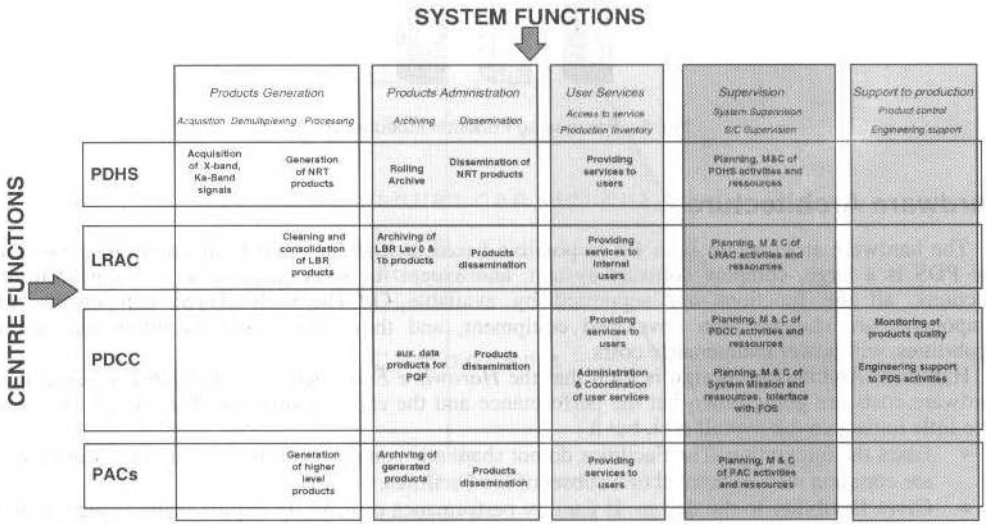


Fig. 4 Identifying independent functions.

This approach has helped to identify the independent functions, that were then mapped on facilities. When the independent functions have been identified, the feasibility of instanciating them (scaleability and configurability) in the different PDS stations and centres and the genericity of interfaces have been investigated, and when this feasibility has been proven, the allocation of the functions to a facility has been performed.

Because the PDS stations and centres missions are different, one independent function cannot be instanciated in the same way in all the stations and centres. The ways of instanciating a facility that have been retained are *scaleability* and *configurability* because they preserve the generic nature of the facility :

Scaleability means that the H/W architecture can be set according to the needs of the station/centre needs. But the same functions are retained and performed by the same facility components.

Configurability only those components needed for a function in a centre, are implemented in that centre.

The fact that an independent function is instanciated also means that it uses different input data, and generates different output data. For example, the archiving function in a PDHS ingests near real time (NRT) data from all the instruments and output Fast Delivery data (FD), and products on media to the LRAC and PACs. However, the archiving function of a PAC ingests data on tape mainly and generates products on tape for PDS external users.

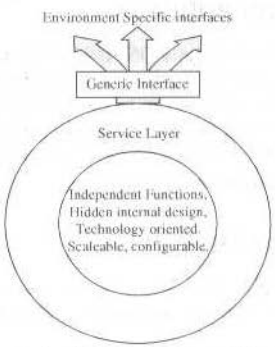


Fig. 5 Principles for Design of Facilities

The input data or output data have all been integrated within generic interfaces (see figure 5), implemented by UNIX standard exchange mechanisms.

The same structure is applied to all ENVISAT Products, defined as an UNIX file with pre-defined Main Product Header (MPH) and Specific Product Header (SPH). Therefore all the products handled by the PDS stations and centres are handled in the same way.

The definition of facilities

The design of the facilities comes from the system analysis, the design constraints, the design drivers and the specific constraints for scalibility and configurability.

These are summarised on figure 6 which shows a shaded block for each main system function. Each system function block contains the facilities with the technology associated with each facility.

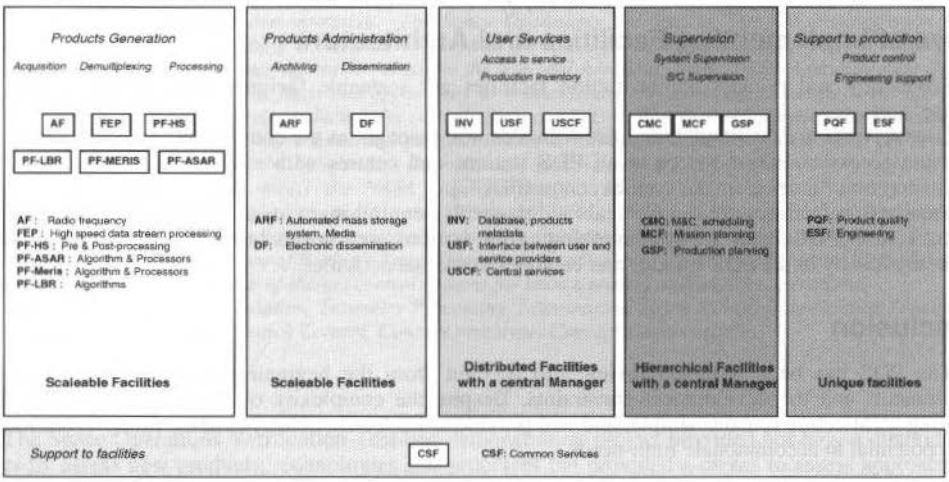


Fig. 6 PDS facilities definition.

The *acquisition function* is implemented in a Acquisition Facility (AF), oriented on Radio frequency technology.

The *instrument packet demultiplexing function* is implemented on the Front End Processor facility (FEP), oriented on high speed data stream (100 Mbps) processing technology. The FEP processes the data stream in order to restore instruments data packets.

The *processing function* is implemented in eight facilities : PF-HS, PF-ASAR, PF-MERIS, AATSR, GOMOS, RA2/MWR, MIPAS, SCIAMACHY. The PF-HS is the host structure that receives the instrument data packets, routes them to the appropriate processor, and interfaces the archive for archiving/retrieval of products or auxiliary data. The other facilities are the PF-ASAR, PF-MERIS, AATSR, GOMOS, RA2/MWR, MIPAS, SCIAMACHY also called Instrument Processing Facilities (IPF). The last five facilities correspond to instruments operating only in low bit rate and are also named under the generic term PF-LBR.

The *archive function* is implemented in the Archive Reference Facility (ARF), oriented on media technology and mass storage/automated library technology and is a service provider for archiving/retrieval. The ARF is the unique media entry/exit point for each PDS station/centre. The ARF storage volume and processing power of the servers are scaleable according to the needs of the PDS station/centre.

The *dissemination function* is implemented in the Dissemination Facility (DF), oriented on electronic dissemination technology. The DF is scaleable according to the PDS needs of the station/centre. The following can be chosen : transmitter, receiver functions and channels used for dissemination (2 Mbps, 64 Kbps links).

The *user access to services function* is implemented in the distributed User Services Facility (USF), coordinated by the User Services Coordination Facility (USCF). The USF is oriented on technologies such as the WWW server and provides user's tools such as product and map visualisation, and documentation.

The *production Inventory* is implemented in the Inventory (INV) which is oriented on distributed Data Base technology and is a service provider for product catalogue information retrieval and update.

The *supervision of station and centres function* is implemented in the Centre Monitoring and Control (CMC) facility which is oriented on SNMP technology for monitoring and control of applications and equipment.

The *system scheduling and control function* is performed by the Monitoring and Control Facility (MCF) and the Ground Segment Planning facility (GSP), based on planning by constraint technology.

The *product control function* is implemented in the Product Quality Facility (PQF) which is a unique facility oriented on product expertise.

The *engineering support function* of the PDS is implemented in the Engineering Support Facility (ESF) which is oriented on system logistics and training.

The last facility (CSF), is not related to system functions, but is the result of design and development considerations. The Common Services Facility (CSF) regroups all the components that are common to several facilities. It is providing UNIX " middleware " services.

Ways of Implementing Facilities and Architecture

Production and product administration facilities are scaleable facilities : processing power, disk storage.

User services are distributed facilities with a central manager as the constraints are to provide to users the same access to PDS services in all PDS stations and centres with a coordination of services and administration of users from the control centre (PDCC).

Supervision facilities are hierarchical facilities with a central manager as the constraints are to have a control centre (PDCC) for the system including the stations and centres but still to retain some level of autonomy locally in the PDS stations and centres for internal activities.

Conclusion

The PDS has been designed taking into account from the beginning of the project the system requirements and the development constraints. Despite the complexity of the ENVISAT mission, the resulting architecture remains simple, and will provide to ESA a flexible system for future evolution with groth potential to accommodate futur needs.

References

ENVISAT-1 Mission and System Summary Is. 2 (ESA)

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp.358-365

ISSN 0100-7386
Printed in Brazil

Space Operations Verification Test-Bed Demonstration

Larry Culver
The Boeing Company
555 Gemini
Houston, Texas 77058
larry.e.culver@boeing.com

Abstract

Boeing Space Operations, a wholly-owned subsidiary of the Boeing Company, has prepared a live Space Operations Verification Test-bed prototype demonstration. The Space Operations Verification Test-bed demonstration was a significant step toward establishing a verification test-bed for the evaluation of the proposed approaches and technologies to dramatically reduce operations costs for the National Aeronautics and Space Administration (NASA). The demonstration also formed the basis and opportunity to provide/share this same infrastructure with commercial space operations customers, thereby reducing the cost of space operations even further. Finally, the demonstration served as the initial phase of a more advanced program where a full-scale Shared Service Center (SSC) can be developed. Specific objectives of the Independent Research and Development (IRAD) project were to : (1) verify the proposed approach/architecture meets the NASA requirements, (2) address the commercial market to satisfy alternative mission and system concepts, and (3) provide an approach to evolve current operational missions to the proposed architecture, which employed commercially-available technology advances in space and ground systems. This paper will present an overview of Boeing's Space Operations Verification Test-bed and illustrate its application in migrating existing systems to state-of-the-art control systems for launch control and satellite operations.

Keywords: *Ground Systems, Simulation, Telemetry Processing, International Space Station, Tele-Science, Commercial Space, Spacecraft Operations, Launch Control, Concept prototype, Concept Demonstration*

Introduction

The Space Operations Verification Test-bed demonstration project provided not only a methodology to rapidly assess new products, technologies and processes but provided a means to assess approaches to the re-engineering of the current procedures and processes. The project also provided Boeing the ability to evaluate the current hardware and software applications for inter-operability, inter-connectivity and ease of integration, and develop a plan for subsequent system deployment. To that end, the Space Operations Verification Test-bed established a new way of doing business.

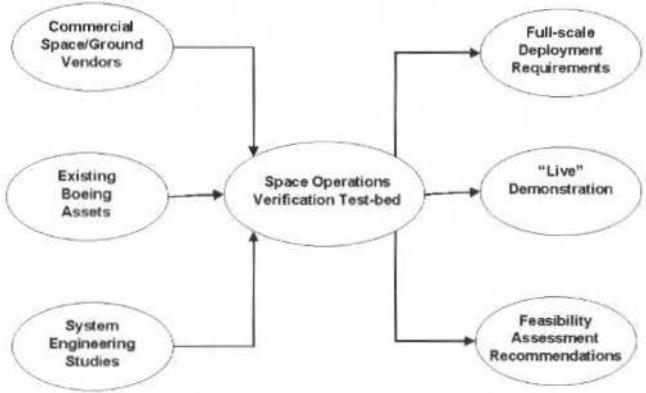


Fig. 1 Space Operations Verification Test-bed

The Space Operations Verification Test-bed is a hands-on evaluation environment patterned after proven systems engineering, development and evaluation processes, which includes advanced ground systems and simulations, and applies open-architecture approaches that permit commercial-off-the-shelf (COTS) hardware and software to work seamlessly across networked users. These methodologies have been applied to construct numerous Alpha (verification) and Beta (validation) sites within Boeing and for external customers. This approach has proven to be very effective, both in the reduction of cost and schedule risk for many programs including Space Shuttle, B-1B, X-31, and assorted Strategic Defense

"Star Wars" projects. This reduction in test beds is achieved in part by the planning, coordinating, and ultimately "leveraging" the resources of these diverse programs towards common architectures and a high degree of reuse of existing Boeing assets and products.

Goals and Objectives

In order for the NASA to meet the nation's future needs for space operations, the cost of operating the existing space and ground systems must be significantly reduced. The Space Operations Verification Test-bed demonstration was an internally-funded IRAD project supporting a six-month study contract to assess approaches and solutions for reducing the cost of operating existing and future spacecraft and ground systems for the NASA.

The purpose in developing the "live" concept demonstration was two-fold. The first goal was to establish a verification test-bed for the evaluation of technical approaches and assumptions to provide risk reduction for subsequent full-scale implementation. Secondly, the Concept Demonstration helped to clarify the architectural and performance requirements for subsequent full-scale development and deployment.

We set out to accomplish these goals by designing and integrating a "Proof-of-Concept" architecture employing commercially-available technology advances in space and ground systems in order to reduce development and subsequent maintenance cost. We used this architecture to demonstrate how shared mission and data services are delivered in order to evolve current operational missions to the new space operations architecture to reduce transition risk. The ability to operate and maintain the current legacy space operations service elements safely and reliably, while in parallel consolidating, commercializing and privatizing these services is an essential element to reducing the cost of space operations and provides the basis for shared, commercially-viable space operations services.

Methodology

The approach or methodology, as depicted in Figure 2, utilized the prioritized process and technology needs from the Study Team's assessment of the NASA's existing space and ground systems. Coupled with the requirement to assess the feasibility to support several operational scenarios or user processes, a set of "demonstration requirements" were defined and baselined. These scenarios included several existing and future NASA space missions, including launch, robotic (unmanned) spacecraft and International Space Station experiment tele-science operations.

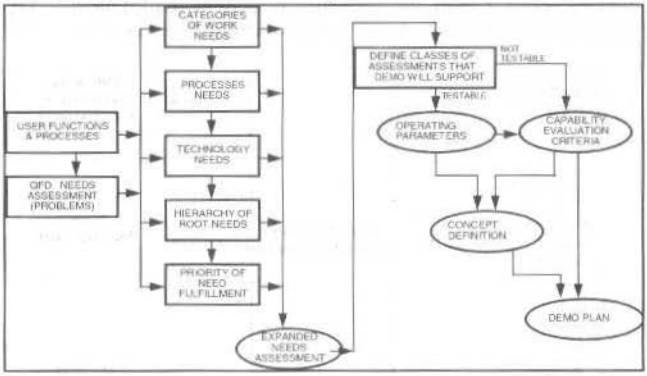


Fig. 2 Needs Assessment Process

In parallel, assessments of the technical and financial performance of alternative "commercial" space operations elements against existing core Boeing capabilities were conducted. These assessments included commercial vendors for computer systems, command/telemetry ground systems, internetworking and communications and tracking systems. Boeing assets utilized included the capabilities for spacecraft simulation, ground systems, International Space Station science experiments, and launch vehicle command/control systems.

All of these elements were integrated into a Independent Research and Development Plan that included requirements definition, concept definition, architecture and implementation plan/cost/schedule. The entire demonstration implementation schedule spanned seven (7) months through the first live demonstration in January 1998.

Demonstration Elements

The Space Operations Verification Test-bed demonstration, as summarized in Table 1, consisted of five major elements or components: (1) Spacecraft Simulation, (2) Simulated Spacecraft Launch, (3) "Live" Satellite Telemetry Data Acquisition, (4) International Space Station (ISS) Experiment Utilization, and (5) Network Management. Each of these major elements combine to provide a 'live", real-time demonstration of the results of the Space Operations Verification Test-bed technology assessments performed by Boeing during this IRAD project.

Table 1 Space Operations Verification Test-bed Elements

Demo Features	Benefits(s)
Spacecraft Simulation ❑ Common simulation infrastructure for pre-flight, concept exploration, design evaluation, development and operations	❑ Reduced life-cycle costs for spacecraft design, development, test and operations
Spacecraft Launch ❑ Boeing Delta 2/3 ACCS through commercial launch facility	❑ Re-use of proven launch control ground systems reduces cost ❑ Sharing of commercial launch facilities reduces cost
Real-time Satellite Data ❑ Reconfigurabe COTS TDM & CCSDS Front End ❑ Leased LEO antenna & tracking systems	❑ "Lights-out" ground station operations reduces cost ❑ Common architecture for multiple missions reduces ops cost
ISS Experiments ❑ Java/Web-based remote Graphical User Interface to ISS experiments ❑ Broadcast experiment video to Investigator	❑ Decreased ISS crew participation in experiment operations ❑ More science
Network Management ❑ Commercial Network Operations Center	❑ Standard, commercial proven tools & interfaces eliminates redundancy and reduces cost and improves reliability

The simulation of the Global Positioning Satellite (GPS) on-board systems demonstrated the capability to train spacecraft ground control operations personnel to react to off-nominal or malfunctions on the spacecraft from any location on the demonstration network.

The simulated launch of a Delta rocket demonstrated the capability and flexibility of the Advanced Command/Control System (ACCS) system to carry out a launch from either the Spaceport Florida complex or a remote control center located in Houston.

The acquisition of real-time time-division multiplexed (TDM) and Consultive Committee on Spacecraft Data Systems (CCSDS) telemetry from low-earth orbit (LEO) spacecraft at the Universal Space Network ground stations demonstrated the capability to consolidate multiple control centers into a single architecture, and the effective out-sourcing of specific services to commercial enterprises.

The ISS Experiment command/control and data acquisition demonstrated the effective application of current internet/web technologies into the ISS to reduce recurring operations costs while simultaneously expanding science data availability.

The management of the demonstration network by an outside commercial vendor demonstrated how effective cost reductions may be achieved through consolidation and out-sourcing.

Demonstration Architecture Overview

Figure 3 provides an overview of the "live" demonstration architecture. This overview of the Demonstration shows that the architecture was geographically distributed across the continental United States and Alaska. The major elements and locations included :

- Ground stations in Alaska and Pennsylvania
- A Network Operations Center in Columbia, MD
- A launch control center complex at Spaceport-Florida
- A simulation complex in Downey, California
- An International Space Station experiment lab in Huntsville, Alabama
- A remote control center in Houston, Texas

It is the flexible network architecture that enabled us to rapidly and easily provide demonstrations in various locations. All monitoring and control of these systems were transferred to the remote control center (located in Houston) for the original demonstration.

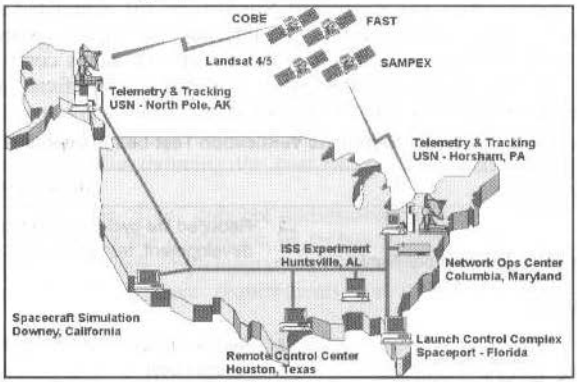


Fig. 3 Demonstration Architecture Overview

Spacecraft Simulation Architecture

The Spacecraft Simulation architecture is based on and utilizes Boeing's Aerospace Simulation and System Test Center located in Downey, California. The Simulation and System Test-bed is also utilized for development of real-time simulations and models, flight software and avionics systems integration, and spacecraft/aircraft ground test systems. For the purposes of this demonstration, we utilized an already existing simulation of the Global Positioning Spacecraft (GPS) on-board systems on the Satellite Systems Test-bed workstation. This simulation test-bed provides the foundation for early, incremental spacecraft system design verification and transition to subsequent test and operational phases.

Fig. 4 GPS Operator-Trainee

This demonstrated the capability to perform collaborative training of ground systems operations personnel using this architecture. The instructor injected faults into the simulated gyroscopes, causing the spacecraft to lose attitude control. The operator-trainee, located in Houston (see Figure 4) was expected to perform commanding of the spacecraft in order to stabilize the spacecraft caused by the failure. Two-dimensional and three-dimensional data visualization on the Remote Control Center workstations is based on commercial products interfaced to the spacecraft simulation over the wide-area network.

Launch Architecture

The launch control portion of the Space Operations Verification Test-bed demonstration is based on the Boeing Advanced Command and Control System from the Boeing Delta II (See Figure 5) Expendable Launch Vehicle Program. Although no rocket was actually launched for the demonstration, a "live" Delta II telemetry stream from a previous launch was played back from the Spaceport Florida launch complex 20. This Delta II launch vehicle telemetry stream was interleaved with an Inertial Upper

Stage (IUS) spacecraft telemetry stream to produce a composite launch vehicle and payload telemetry stream.

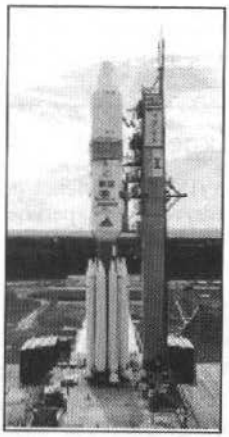


Fig. 5 Delta Launch Vehicle

This demonstrated the capability to carry out a launch utilizing commercially-available hardware and software products. The launch range facility and services were out-sourced to Spaceport Florida Authority at the Cape Canaveral Air Station. Although the launch control complex in Florida controlled the simulated launch, the launch was monitored from the Remote Control Center in Houston. This Remote Control Center replicates the workstation hardware and software, and is fully capable of carrying out the launch from this location. All two-dimensonal and three-dimensional data visualization and display was performed using commercially-available software, tailored to provide displays unique to the IUS payload and the Delta II launch vehicle. This reduces the cost of developing and maintaining these systems.

Real-time Spacecraft Telemetry Data Acquisition

During the Real-time Satellite Telemetry Data Acquisition portion of the demonstration, we tracked and acquired real-time telemetry data from five existing, operational spacecraft: FAST, COBE, SAMPEX, LANDSAT-4 and LANDSAT-5. Each of these low-earth orbiting (LEO) satellites were tracked from the Universal Space Network (USN) commercial ground stations in Alaska and Pennsylvania. All of the ground systems telemetry processing hardware and software is commercially-available and allowed us to process both Time-Division-Multiplexed (TDM) and CCSDS (Consultive Committee on Spacecraft Data Systems) telemetry formats through a single architecture (See Fig. 6).

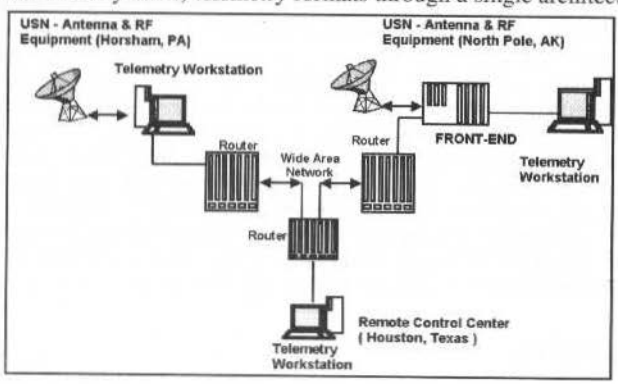


Fig. 6 Real-time Satellite Telemetry Data Acquisition Architecture

Spacecraft ephemeris and ground station access predictions were generated using commercial-off-the-shelf software. The "lights out", flexible design and the ability to easily re-configure the ground

station telemetry processing systems enable us to consolidate and eliminate satellite-unique control centers and ground systems. The capability to operate these systems in parallel with existing, operational ground systems for each of these spacecraft enables us to mitigate risks and decrease costs in the transition to the new ground systems. The use of the Universal Space Network commercial ground stations enables us to share these facilities with other USN customers, thereby reducing the cost to operate these systems even further.

International Space Station Experiment

During the International Space Station (ISS) tele-science portion of the demonstration, we utilized an existing prototype ISS Cell Culturing/Bio-reactor experiment located in the Boeing laboratory in Huntsville, Alabama. The Cell Culturing test-bed in Huntsville (depicted in Figure 7) was controlled from a Remote Control Center in much the same fashion as an orbiting experiment might.

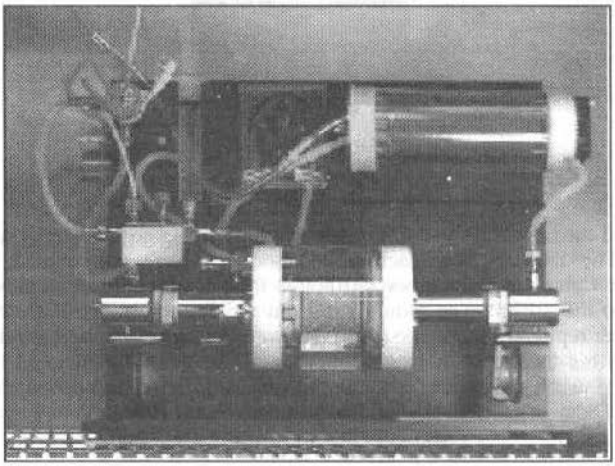


Fig. 7 ISS Cell Culturing Experiment

This internal research and development project is pat of an on-going effort by Boeing to address some of the many new challenges to achieving cost-effective utilization of International Space Station resources. These challenges can be largely expressed as increased demands for resources which can be categorized in three groups, the increased demand for communications bandwidth, requirements for crew participation in experiments, and the ground support services required to conduct world class scientific research.

Tele-operation, a key ingredient to space operations, allows a remote experiment to be an extension to the scientists very own laboratory. Tele-operation allows the scientist to perform real time action based on the current experiment status and experiment results. Depending on total automation, or if participation by a trained crew member is required, this removes the flexibility from the scientist which is often necessary to achieve real scientific breakthroughs and get the most science for the money on a given flight. It is also well known that even with crew member presence, such as on the space shuttle or International Space Station, the crew member's time is severely limited for any one given experiment.

This portion of the demonstration verified our capability to enhance the scientific return from space borne experiments through effective application of Java/web-based user interfaces that enable the principal investigator, located on the ground, to participate more fully in the conduct of their experiments. Other than the experiment-unique Java applet interfaces operated through a web browser, all of the hardware and software is COTS. A remotely-operated video camera and PC-based video conferencing software enabled the scientist to view the experiment in real-time as the experiment ran.

Network Management

The Network Management portion of the Space Operations Verification Test-bed demonstration utilized a commercial network operations center located in Columbia, Maryland to perform monitoring/fault detection, configuration management and performance tracking of the demonstration network components.

The Network Operations Center (NOC), depicted in Figure 8 has the capacity to manage up to 150 Fortune 500 sized networks, equivalent to 250,000 end users. The network management tools are a set of market-driven, best in class technologies that assure compliance to real and de facto standards and reduce the risk of obsolescence. These tools were utilized remotely during the demonstration to manage all elements of the demonstration network, including wide-area network circuits, routers, local area networks and individual workstations.

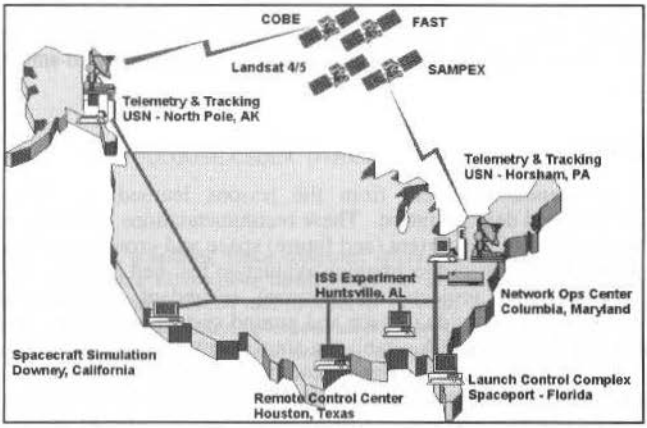


Fig. 8 Space Operations Infrastructure Network Architecture

This portion of the demonstration verified that out-sourcing of the network management of the ground system elements reduces operations costs, improves reliability and increases the visibility into the network operations. This is accomplished by reducing the number of interfaces, consolidating similar functions, and by standardizing on well-proven commercial tools and practices.

Summary and Conclusions

The methodology used to develop the Space Operations Verification Test-bed "live" demonstration was driven by our goal to address the need to assess approaches and technologies to reduce life-cycle costs for the existing space and ground systems. This demonstration was completely assembled and integrated in approximately seven (7) months using Boeing discretionary Independent Research and Development funds to provide a "live" concept prototype and technology verification test-bed.

This project has afforded Boeing the opportunity to assess the commercial space and ground systems providers to satisfy current and future space operations technical and cost performance requirements.

Furthermore, the establishment of a single test-bed for the entire spacecraft development life-cycle (prototyping of system concepts, design, development, test and operations) provides leverage in the form of a common infrastructure or architecture for a diverse set of programs, reducing the cost and risk to deploy advanced ground systems technologies.

The "out-sourcing" of specific portions of the architecture to commercial vendors or service providers whose business is based on these markets reduces the implementation risk and recurring operational costs. In particular, as more spacecraft operators utilize the commercially-available telemetry and tracking systems, the cost for operating and maintaining these systems is significantly reduced through sharing of the cost burden among many customers.

The elimination of replicated or redundant functions through consolidation increases reliability and visibility, while simultaneously reducing operational costs. This was particularly significant with regard to the management of the disparate, and geographically-distributed network components and workstations. In addition, the use of common, open-system interface standards enabled the rapid component integration and flexible network architecture.

The replacement and consolidation of in-house, "institutional" capabilities with COTS tools and software reduces operations and maintenance cost.

Of particular importance to us is the replacement of spacecraft-unique control centers with a single, COTS product-based control center and ground system architecture that supports many existing, operational missions. The Space Operations Verification Test-bed enabled us to deploy and operate this new architecture in parallel with the existing operational systems, reducing the technical risk and cost of

implementation. Thus, this verified the approach and technology required to migrate these existing space missions to a common architecture for control center operations.

The use of a common simulation infrastructure or architecture across the entire spacecraft development and operations phases reduces the spacecraft life-cycle cost. The Space Operations Verification (simulation) Test-bed provides the capability to rapidly prototype operational concepts, design and verify new spacecraft components and train personnel on spacecraft operations.

The concept prototype demonstration formed the basis for the definition of the requirements for the full-scale Shared Service Center. The concept demonstration also permitted the "hands-on" evaluation of the latest commercially-available hardware and software technologies to meet the NASA mission support requirements.

Recommendations

The following recommendations stem from the lessons learned by implementing the Space Operations verification Test-bed demonstration. These recommendations are being made to the NASA to reduce the cost for the operation of the current (and future) space and ground systems.

- NASA should establish a shared technology evaluation test-bed (Shared Service Center) that can be utilized by multiple projects to eliminate redundant procurements and reduce the implementation costs for individual space and ground system projects or programs.
- It is recommended that the NASA establish a single control center architecture based on proven, commercially-based hardware and software components for all spacecraft.
- NASA should utilize commercial ground station providers to out-source the ground station operations, permitting the operations cost to be distributed among the many government and commercial spacecraft operators.

Acknowledgments

I wish to acknowledge the participation by many individuals and organizations, without whose dedication and support this project would not have been possible. The team effort demonstrated by these individuals from the various organizations and companies was exemplary.

Although it is impossible to enumerate everyone who participated in the implementation of this project, I want to especially thank these individuals:

- Boeing - Paul Schoen, Ken Stern, Joe Gernand, Kellie Miller, Romeri Jakpor, Velda Posada, Stephen Price and Kenny Allen;
- Colsa Corporation - Mark Emery and Carlos Garcia;
- Universal Space Network - Bob Weaver and Mike Steckel;
- Lucent Technologies - Joe Baleno;
- Digital Express - Andy Haug;
- Integral Systems - Don Johnson;
- TSI-Telsys - Kathy Jones;
- Science Applications International Corporation -Fabrizio Pela

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp.366-369

ISSN 0100-7386
Printed in Brazil

INTELSAT FDC : A Breakthrough in Satellite Operations Automation

Jean-Michel Darroy
Operational Centers
Test Systems and Simulators Division Manager
jean-michel.darroy@tls.mms.fr

François Copin
FDC Project Manager
MATRA MARCONI SPACE
31, Avenue des Cosmonautes 31402 TOULOUSE CEDEX (France)
francois.copin@tls.mms.fr

Abstract

MATRA MARCONI SPACE (MMS), a leading space company, offers a wide range of ground systems components, from the smallest satcom components to complete turnkey systems. The company's capability spans engineering, development, integration, validation, installation and operations support, for ground stations, satellite control and mission centers, satellite simulators, test facilities, communication networks and systems.
MMS is performing, through programs and internal funding, extensive work in the area of ground systems and operations preparation and automation. This work has led to MMS products that can support the needs in term of automated operations, for both individual spacecraft and large satellite fleet or a constellation : namely the OPEN CENTER components suite, the OPSWARE operations products suite and the QUARTZ++ Flight Dynamics System.
MMS was awarded a major contract from INTELSAT for its Flight Dynamics and Commanding system (FDC). The main characteristics and requirements of the INTELSAT FDC are to conduct operations of a large fleet of satellites within a distributed and multi-user environment with the key objective to dramatically reduce costs through operations automation.
This presentation will deal with MMS' FDC system and will focus on its automated operations features.

FDC Project Overview

The INTELSAT fleet of communications satellites is one of the world's leading satellite communications systems. It provides voice, data and video services to over 200 countries and territories. During the 1998 World Cup soccer championship, 11 of INTELSAT'S satellites were used to beam the games to more than 3 billion people.

INTELSAT is developing new satellite operations systems to replace or enhance its current systems.

The FDC system will be capable of operating a fleet of up to 30 satellites of seven different satellite series. The proposed system incorporates Commercial Off-The-Shelf (COTS) items and takes advantage of the experience and skills of Matra Marconi Space (MMS): satellite operations for Launch and Early Orbit Phase (LEOP) as well as for station keeping.

Matra Marconi Space will be using its own in-house software products, mainly the OPEN CENTER and OPSWARE product lines covering all automation levels required by the FDC, its new generation QUARTZ++ product for Flight Dynamics functions and the NetExpert product from OSI, which implements the remote device management monitoring and control of the INTELSAT TTC&M stations.

FDC will interface with existing IS ground network systems. The FDC will perform the functions of satellite command, flight dynamics, ranging, ground segment device control and events and alarm management, including :

- Multi-satellite operation
- Multi-user operation with dedicated security and safety administration mechanisms
- Interface with existing components for continued operation during deployment of the system
- Simplifying maintenance by exploiting current technology and client server architecture
- Maximizing the use of ground segment resources through active resource management and automated ranging
- Integrating functionality onto a unified console and platform
- Improving events and alarms management for both satellites and networks
- Increasing flexibility and access during commanding by using INTELSAT's central database for shared information
- Exploiting existing technologies to set the framework for increasing automation

FDC functions ensure the generation, scheduling and execution of satellite commands. These functions include all of the functions that are necessary to operate a reliable and flexible ground support

network. INTELSAT currently operates a large fleet of satellites in Geosynchronous orbit. There are seven different families of satellites, built by several manufacturers, including : INTELSAT V (IS-V), INTELSAT VI (IS-VI) INTELSAT VII (IS-VII) INTELSAT K (IS-K) INTELSAT VIII (IS-VIII) INTELSAT IX (IS-IX) and K-band Television (K-TV). All of these satellites share the TTC & M network made up of six stations worldwide. The FDC System will provide the ability to operate the satellite fleet from INTELSAT headquarters. Satellite commanding shares the same resources as Ranging and other communication monitoring systems. The goal of FDC is to manage these functions and utilize the resources in an efficient manner.

The central idea of the FDC is **the concept of a Session**. A Session is an entity that allows a user-defined work Activity, such as Commanding or Ranging, to obtain the resources necessary to accomplish the work. For example : Command Sessions obtain the resources to transmit satellite commands to the satellite, Ranging Sessions obtain the necessary resources for the work to be done by the ranging system. The Session allocates the resources (Services), is inserted into the schedule, and when the Session Start Time arrives, initiates the network commands necessary for the set up of the resources required to carry out the work of the requesting Activity. Conflicts for resources are handled automatically if possible, with manual intervention if necessary or desired.

Operations of the INTELSAT satellite fleet includes stationkeeping, responding to customer configuration changes, satellite health maintenance, eclipse operations, anomaly investigations, and transfer orbit operations. All Command Activities, whether generated manually or automatically, can request the creation of a Command Session.

To initiate satellite commands manually, a satellite engineer has several existing tools to assist him. The FDC will utilize these existing tools to allow the engineer to define Command Activities that describe the satellite commands and possibly some control logic associated with those commands. An engineer can, after review and approval, then request the FDC to create and schedule a Command Session for the Command Activity.

Similarly, automatic software functions, triggered by maneuver messages from the Flight Dynamics function of FDC, by external demand for satellite configuration changes, or by an internal satellite diagnostic facility, will generate or select Command Activities and requests to create and schedule Commanding Sessions.

Ranging Sessions are generated in response to manual demand, or automatically based on maintenance of predefined ranging schedule rules. Similarly, Communications Monitoring Sessions may be requested by external systems.

All of these sessions will have Services associated with them that will define ground-based devices that must be configured , controlled an monitored to allow the Session's client activities to be performed. The FDC will ensure the availability of the services, allocate them to the Session, and insert the session into the schedule. All users of the system will be able to monitor the scheduled sessions through the Timeline display.

The Display of the Schedule (Timeline) Will be the Key Means to Operate the Satellite System

Graphical User Interface (GUI) drill-down techniques will provide all of the necessary information concerning Scheduled Sessions. For example : the services allocated to the session will be visible through block diagrams with indications of their health and status. Modifications to the schedule by operators will be possible, with proper controls, through the Timeline display.

When Scheduled Session's start time reach the current time, the FDC will issue the necessary network commands to activate a session that ensures that the network devices required by the client work activities are configured and available. The client retains control of the devices until the activity is completed. At termination of the activity, the resources are released and become available for other sessions.

Once a Session is active, the client activity can execute the functions necessary to perform the work. For example : Command Activities can now transmit and execute the satellite commands defined. The potential for hands-off operation of Command Activities will exist. The FDC defines extensive functions that will be available for the Satellite Operations staff to monitor and interact with Command Activities. This flexibility will allow operators to respond much faster to anomalous conditions.

The scheduling of Ranging Sessions is an additional feature of the FDC. FDC will schedule Ranging Sessions automatically under a wide variety of conditions.

The Definition of a New Flight Dynamics System that Uses the Ranging Data and Provides the Maneuver Messages to Derive Automated Commands Generation is Another Key Operational Concept of FDC

Providing timely orbit determinations and scheduling of maneuvers is an important part of satellite operations. The ability to manage events and alarms, of all satellites and the ground network will be provided by FDC. Situational awareness by controllers and engineers will be accomplished by using the flexible event and alarm facilities defined for FDC. Events and alarms functions will be supported by on-line procedures and graphical user interfaces.

FDC Architecture and Major Components

The FDC system is based on world class products and state of the art technologies (see table 1)

- MMS Operations products
  - OPSWARE/OPSAT and OPS-EXECUTER for operations procedures preparation and automated execution
  - OPSWARE/TIMELINE for operations scheduling preparation and automated execution
  - OPEN CENTER for commanding services
  - QUARTZ ++ for flight dynamics
- OSI Network Management product
  - NETEXPERT
- Others products
  - IONA/ORBIX for CORBA middleware implementation
  - ILOG/SOLVER for resources management and optimization
  - ILOG/VIEWS and NEURON DATA/ELEMENT PRESENTER for graphical users interfaces
  - ORACLE data base management system
  - IS Systems

Table 1 INTELSAT FDC Software Subsystems

FDC Function	FDC Subsystem	Associated Product	Provider
Satellite Commanding	ACP : Automatic Command Procedures	OPSAT	MMS OPSWARE
		OPSEXECUTER	MMS OPSWARE
	IIC : Interactive Immediate Commanding	Open IIC	MMS Open Center
	Commanding Services	Open Command	MMS Open Center
	Autonomous Satellite Management Syst.	Specific	MMS Open Center
Device Control Management (DCM)	DCM Schedule Manager	TIMELINE EXECUTER	MMS OPSWARE
	DCM Network Manager	NetExpert	OSI
Flight Dynamics	FDS Kernel	QUARTZ++	MMS
	FDS Schedule Manager	TIMELINE	MMS OPSWARE
Ranging System	Ranging	Specific to INTELSAT	
Support Services	Event and Alarm System (EAS)	NetExpert	OSI
	Databases	Oracle	Oracle
	Access Control	ACC	MMS Open Center
	Application monitoring and Control	AMC	MMS Open Center

The use of the above mentioned products and COTS Products minimize custom software development. This will also allow INTELSAT to take advantage of product and COTS upgrades during the lifetime of the FDC system.

MMS World Class Products are Already Operationally Used by Many Other International Customers

As an example, OPSWARE products are used, all over the world for satellite operations automation, by SES (ASTRA Satellites), WORLDSPACE, NILESAT, HISPASAT, ST1,...

OPSWARE, OPEN CENTER and QUARTZ ++ products are respectively described, with more details, in three different papers which are also presented in this SCD II conference.

The System is Designed With Stringent Requirement on Availability

24 hours a day, 365 days per year operations. The architecture is based on a redundant hardware and software architecture to ensure high availability of headquarters (HQ) functions. All applications can restart from a context/checkpoint. Contexts are also saved through Oracle records, and therefore relies on DB mechanisms and in particular DB replication.

FDC Project Product Releases

FDC project is a very challenging project because of its ambitious technical and operational objectives.

Three successive Product releases are planned :

- R1 (version 1 of the Flight Dynamics Systems)
- R2 (version 1 of the full FDC system)
- R3 (an enhanced version of the full FDC system)

It is anticipated that these objectives can be met thanks to the use of existing products and COTS, and also thanks to an intensive concurrent engineering work jointly performed by INTELSAT and MMS once the project was kicked-off.

Conclusions and Perspectives

The INTELSAT FDC system opens a new era in the domain of satellites operations, characterized by:

- Operations Automation
  Satellites engineers will be able to focus on satellite operations and will thus be relieved from day to day operations.
  Satellite controllers will be able to focus on important operations through a situational awareness.
- Schedule and Resources Optimization
  Operations can be scheduled ahead in time with an optimized resources allocation scheme, and then be executed automatically.
  An improved and more global view of the ground segment usage is thus now provided.
- TTC & M Sites Remote Operations
  Reduced operation staff at the TTC&M sites.
  The FDC system, and its various building blocks, will be used, and thus labeled "flight proven", for satellites built by the major international satellite manufacturers.

Aknowledgement

This MMS paper is hereby presented based on the project work made possible thanks to the talent, the experience and the commitment of both INTELSAT and MATRA MARCONI SPACE teams.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 370

ISSN 0100-7386
Printed in Brazil

Data Processing Systems III

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp.371-378

ISSN 0100-7386
Printed in Brazil

Are Standards Cost Savers? What are the Benefits of SLE-Services for the Users?

Martin Pilgram
Deutsches Zentrum für Luft- und Raumfahrt e.V. (DLR)
martin.pilgram@dlr.de

Abstract

Overall costs of a spacecraft mission have to take into account not only satellite costs but also costs of the ground segment. Within the ground segment costs can be driven by the ground segment infrastructure or the operations using this infrastructure. From a ground system view satellite operations that use only one ground station, in the most simple case the user's ground station, will be a cost effective solution. But today more or less all small space agencies need cross support for their satellites, at least in the LEOP phases of their missions.

One way of reducing cost in using cross support is to standardize those interfaces. Therefore the space agencies asked their standardization organization (CCSDS) to define services that give the owner of a spacecraft the ability to operate his spacecraft through different ground stations in a standardized way. This paper summarizes the status of these definitions and the future steps. It shows how the users can benefit from these definitions and demonstrates the cost savings of such an implementation.

Keywords: *spacecraft operations, standards, CCSDS, Space Link Extension (SLE) Services, cost reduction, cross support*

Cross Support and standards in DLR/GSOC

Small agencies without a global network of ground stations have to use other agencies ground stations at least for LEOP phases of their missions. Therefore it is of big interest for a agency like DLR to get standardized interfaces from the network providers (ESA, NASA, ..) to minimize cross support costs.

History of Cross Support

DLR/GSOC's ground station is situated at Weilheim (50 km south of Oberpfaffenhofen) and consists of one 30m, two 15m and one 11m antenna. All these antennas are integrated in the GSOC propriety communication system called „Datanet". This system is based on the DECnet networking protocol using in-house data structures for routing the data between the ground station and the control center as well as inside the system itself.

Fig. 1 Weilheim Ground Station

At least for LEOPs there is a need for a network supporting GSOC.

On the other side GSOC delivered Back-Up services for Eutelsat, Eumetsat and the German Telekom. In the past gateways were implemented to support each of these different providers or users. Today this includes gateways to NASA's NASCOM, ESA's X25-network, and some others as CNES, IRSO,

Eutelsat, Eumetsat, Taiwan and Korean satellites. Sometimes own hardware was positioned at the customers ground station.

Up to today, the interface discussion between the agencies were always based on the implemented propriety systems of each of the agencies. But because nearly every agency is using there own system new gateways have to be implemented whenever a new system should be connected to the GSOC system.

This is not a very effective. But up to now there is no standard defined that can be taken for support a service between the agencies. CCSDS, the Consultative Committee for Space Data Systems, is aware of this and has been triggered by the agencies to generate standards for inter-agency support.

CCSDS

A significant trend exists within national and international space agencies towards using standard techniques for handling space data. By cooperatively developing these techniques, future data system interoperability may be enhanced. Recognizing the benefits an international Consultative Committee for Space Data Systems (CCSDS) was formed in 1982 by the major space agencies of the world to provide a forum for discussion of common problems in the development and operation of space data systems. It is currently composed of ten member agencies, twenty-three observer agencies, and over 100 industrial associates.

Members		
ASI/Italy	ESA/Europe	
BNSC/UK	INPE/Brazil	
CNES/France	NASA/USA	
CSA/Canada	NASDA/Japan	
DLR/Germany	RSA/Russia	
Observers		
ASA/Austria	EUMETSAT/Europe	MOC/Israel
CAST/China	EUTELSAT/Europe	NOAA/USA
CRC/Canada	HNSC/Greece	NSPO/Taiwan
CRL/Japan	IKI/Russia	SPO/Belgium
CSIR/South Africa	ISAS/Japan	SSC/Sweden
CSIRO/Australia	ISRO/India	TsNIIMash/Russia
CTA/Brazil	KARI/Korea	USGS/USA
DSRI/Denmark	KFKI/Hungary	

The goals of CCSDS are:

Establish a world-wide, open, CCSDS-compatible virtual space data system for international cross support, interoperability, and science information interchange.

The benefits of CCSDS are described as:

- promotes understanding of exchanged data
- reduces nonrecurring costs
- fewer project-unique developments
- short system test periods
- less training/retraining of personnel
- reduces recurring costs
- more commercial-off-the-shelf hardware
- fewer facilities because of load leveling
- only selected system redundancy
- more automation
- reduces mission risk
- enables ingest/access to long-term data archives

Today CCSDS work is structured into three working groups, so called panels:

Panel 1: Telemetry, Tracking and Command

Panel 2: Standards Information Interchange Process

Panel 3: Cross Support Operations

CCSDS Recommendations are routinely submitted to the International Standards Organization (ISO) through ISO Technical Committee 20 (TC 20 Aircraft and space vehicles) / Subcommittee 13 (SC 13 Space data and information transfer systems). Many CCSDS recommendations have already been

adopted as international standards, and many others are currently in the review process leading to adoption by ISO.

Information published by the CCSDS is available on the Web at: http://www.ccsds.org/ccsds/

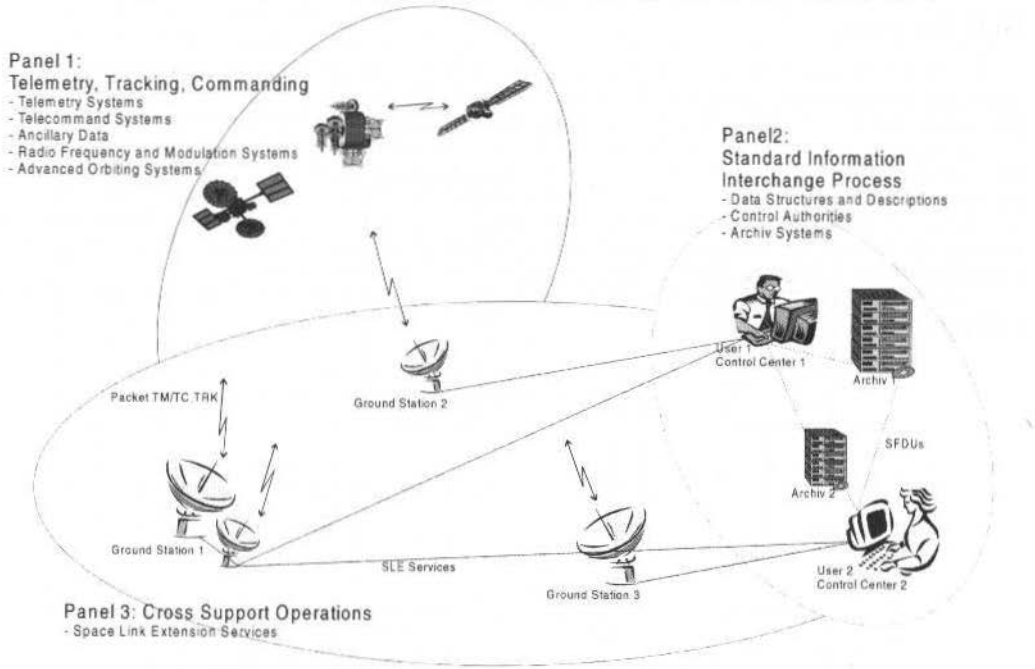


Fig. 2 CCSDS panel structure

Implementation of CCSDS Standards in DLR/GSOC operated spacecraft

More and more spacecraft today start to implement CCSDS standards for packet TM/TC. The following list shows the current status on these implementations from a DLR/GSOC view.

Satellite Name	Abrixas	Champ	Eutelsat	Bird, Grace,...
purpose	scientific X-ray	scientific gravity/magnetic field	commercial telecommunication	
life time	1999-2000	2000-2002	1998-2005	
orbit	low	low	geo	
local G/S	yes	yes	yes	
Cross support	NASA	NASA,	NASA, ISRO	
Data type	HK + SCI	HK+SCI	HK	
data rate Kb/sec	d:500, u:4	5, 32, 1000	2	
Uplink Pkt	yes	yes	yes	
Uplink Frames	yes	yes	yes	
Uplink Code	yes	yes	yes	
Downlink Pkt	yes	yes	no	
Downlink Frames	yes	yes	yes	
Downlink Code	yes	yes	yes	

Fig. 3 CCSDS standards in current DLR operated spacecrafts

Beside these spacecraft other space agencies are asking for support and operation of their spacecraft, which are using at least packet TM/TC. These include Taiwan as well as Korea.

SLE Services

In time, more and more satellites will be using CCSDS standards for receiving telemetry from the spacecraft or sending telecommands to the spacecraft. At the ground station this is getting even more problematic because of the data products a ground station can deliver or accept: packets, frames,..

Another question that occurs is the following: Will CMD retransmission be done at the station or has the user to support this feature in his control center? Looking at all these items, standards seem to increase the cost instead of decreasing it. Why? Is there something missing in the processing line?

A clear definition on how to interface between users and groundstations is needed. This has to include not only the datastreams that are exchanged, but also the protocol that will be used (simple TCP, CMIP, CORBA, DCOM) and management aspects like security or quality of support.

CCSDS put these questions into their standardization line in defining Space Link Extension (SLE) Services.

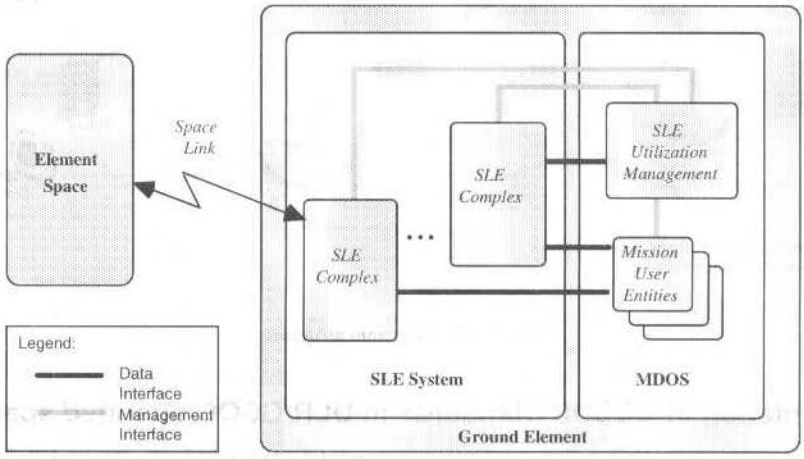


Fig. 4 SLE Complex/MDOS interfaces

The SLE Recommendations complement the CCSDS Space Link Recommendations with a range of services that are required to configure, operate, and supervise the ground data systems. SLE Recommendations apply to data systems that are able

- To receive CCSDS Space Link data structures from a spacecraft via a Space Link, or
- To send CCSDS Space Link data structures to a spacecraft via a Space Link, or
- To transfer such CCSDS Space Link data structures between ground-based entities.

The following definitions are used in the SLE terminology:

Ground Element

The ground element of a Space Mission Data System includes a Mission Data Operation System (MDOS) and an SLE system. It may also contain other components.

SLE Complex

An SLE Complex is a set of SLE-Functional Groups under a single management authority. At the time of cross support, an SLE complex has a single established relationship between the SLE system and the MDOS.

Functional Group

A functional group performs the function that performs an incoming data unit from a SLE data channel of a given type to produce and provide a related group of SLE transfer services. In this functional view, the SLE system is decomposed into SLE Functional Groups (SLE-FGs), which implement and augment the ground side of the Space Link protocols described in CCSDS Packet Telemetry, Telecommand, and AOS Recommendations.

SLE services are provided by a SLE system to an MDOS.

SLE services comprise:

- The **SLE transfer services**, which transfer SLE data channels from/to the space element to/from the MDOS through the SLE system;
- The **SLE management service**, which controls the scheduling and provision of instances of SLE transfer services by the SLE system.

The following two figures show the forward telecommand functional groups and the return functional groups (another section are the AOS forward functional groups).

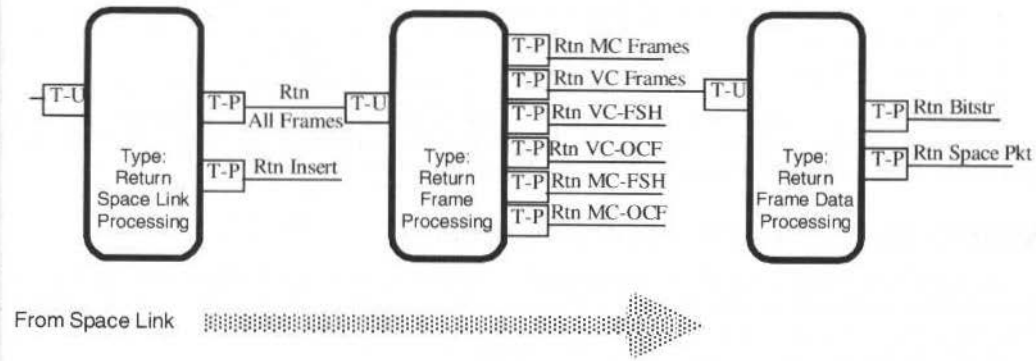


Fig. 5 Return Functional Groups

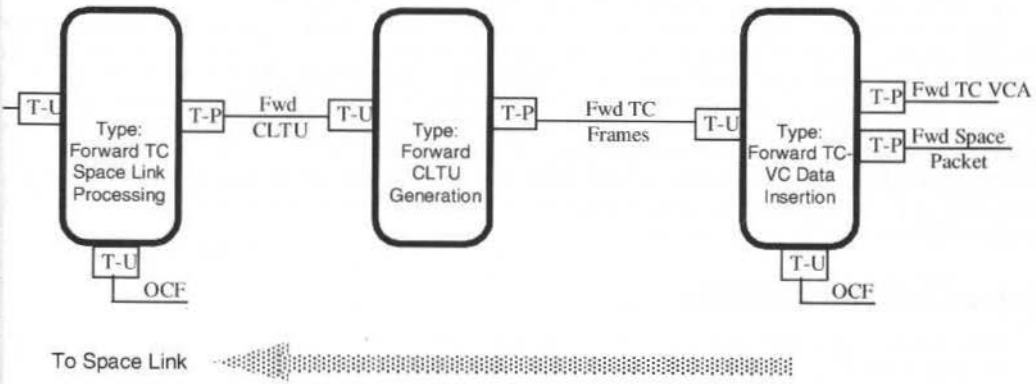


Fig. 6 Forward Telecommand Functional Groups

Status of CCSDS Panel 3 recommendations on cross support

Service Management recommendation will be issued mid-99 as a Red Book, a draft (white) is already available.

In the area of SLE service recommandations the following Red Books (ready for member agency review) are available:

- Return All Frames
- Return VC & MC Frames
- Froward CLTU
- Forward Packet

Four more services books are in a draft status. The goal is to get all the services published as recommendations in 2000.

Beside this, work is being done on SLE application API and on security issues. At least on API there will be a draft in 1999.

CCSDS implementations by baseband providers

The main baseband providers started delivering components, which are said to be CCSDS compliant, in 1998. Some of the products are listed here:

- Avtec (PTP – Programmable Telemetry Processor; Monarch – PCI Frame Synchronizer/Telemetry Simulator with Reed Solomon Encoder/Decoder)
- L-3 Communications (NETstar, IMPACT 2000)
- TSI-Telsys (CSP – CCSDS Service Processing card; CI – Command Interface card; LCCG – Low Cost Communication Gateway)
- Berg Systems (Model 4428-PCI CCSDS Downlink Interface)
- In-Snec (Cortex)
- Enertec (3801-20)
- SIL

But all of these systems have proprietary interfaces for delivering and accepting telemetry, telelcommands on frame, channel or packet base and also for configuring the system. Will the next generation of these system implement SLE-services? We hope!

Need for SLE Services in GSOC (extern/intern)

As mentioned before, DLR/GSOC has to operate in the next years the satellites Champ, Abrixas, Eutelsat, Bird and Grace, which are using part or the full set of these standards. And as DLR/GSOC normally operates in a cross support environment – necessity of an antenna network for a LEOP missions – DLR/GSOC has to implement a first set of SLE services.

This first set of services includes

- Forward CLTU and TC Packet Service
- Return All Frames, MC-OCF, Virtual Channel and TM Packet Service

One aspect of cross support is the interoperatibility with other networks, the other is the location of the own ground station at Weilheim. An 1 Mbit data connection between the ground station and the control center means that, in some cases (e.g. CHAMP), only part of the telemetry can be transferred in real-time from the ground station to the control center. One possible solution to transfer the necessary data stream is to select the „real-time" packets at the groundstation for transferring. This capability is one driver for an implementation of SLE return services.

On the other hand more and more experimenters want to command their experiments on a spacecraft from their home location without knowing what all the other experimenters are doing. This speaks for an implementation of the TC packet service.

Implementation Aspects

For implementation the following main aspects will be looked at:

- Interoperability – what are the other space agencies implementing,
- underlying protocols – is TCP the right solution,
- current technology – is SSL accepted for security issues - and beside this all the
- currently running old system.

The API

The SLE provider and user entities realize the CCSDS ground element services and in particular provide an application programming interface (API) to user applications. This API will be implemented in a three layer approach:

1. Offer an SLE operations interface to the service user application.
2. Implement SLE service behavior common to all operations (e.g. buffering).
3. Implement the interface to the middleware (proxy) and handle the transfer syntax.

The proposed implementation should be:

- Object oriented,
- The three components are dynamically linked.
- The interfaces are reused between components 1 and 2 and 2 and 3.
- A change in middleware requires a new proxy component (and the new middleware)
- Component selection is done at start-up.

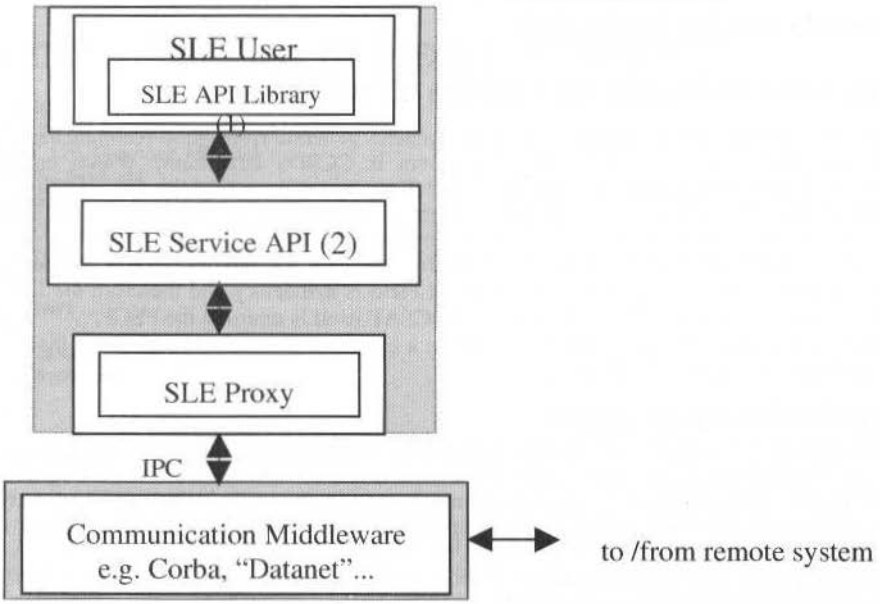


Fig. 7 SLE API implementation structure

Middleware

As seen in the API figure, the first implementation will use an off the shelf CORBA implementation and the old GSOC specific DATANET system for communication functions.

With this approach, we can combine the functionality given by communication packages as CORBA and the connectivity into our old existing system, which uses DATANET for all internal communication.

During first Tests of the implementation the CORBA interface has to be tested in terms of overhead and performance.

Processing Engines

A level-0 processing will be done using the already implemented SW-packages for TM/TC which handle the different CCSDS processing layer (packet, frame, segmentation) .

Testing

For test purposes a independent packet to frame generator is used.

Implementation Path and Time Scale

Because of having to serve the old GSOC DATANET system the implementation has to be based on an interface or gateway into this DATNET system. The first set of SLE proxies that will be generated will serve this interface. A second set of proxies will be a CORBA implementation. In parallel an SLE API library will be defined to serve the main and today defined services. During this implementation no AOS functionality will be incorporated.

A lot of the processing functionality is already developed in a forward and return processing library, which easily could be used to include the processing part into the services. We started implementing this processing functionality last summer for our current missions Champ and Abrixas.

A first version of the services will operate on a very basic management service implementation. That will be of course a more or less hard coded part of the necessary information to provide the defined services.

implemented:	TM, TC library of SLE processing functionality
end 1999:	SLE Services as described
end 2000:	SLE Service Management, if defined by CCSDS in 1999.

Some words on standardisation

Is standardization always the solution (ESA/ROCSAT)

CCSDS develops recommendations. The space agencies generate from these recommendations their own standards. But how can one check if a system is CCSDS compliant? Which options of a recommendation are implemented and which are not? There are still possibilities for a satellite that has implemented packet TC cannot be commanded by a groundstation that has implemented the standard as well. There has to have been the same set of options implemented!

The Taiwan satellite ROCSAT is a good example for this. ESA was not able to handle it because:

- In the ESA Standard the Frame Error Control Field is mandatory and therefore the ESA Packet telecommand Encoder always generates it. ROCSAT itself is omitting the FECF.
- In the ESA Standard each frame is inserted in a corresponding CLTU. ROCSAT inserts several small frames into a single CLTU.

Is standardization a cost factor?

Discussing new gateways for new satellites will always bring up questions by the management like: You did your standardization, but why do you still have to implement all this stuff? The answer is that the functionality that comes with this standard is much more than we had before. And only a full implementation of the standard will give you the peace to stay with your system for a lot of different missions.

From this point of view, not standardization is the cost factor but the new features you have to include in the standards to keep standards accepted beyond tomorrow.

Standards versus new technology

There will always be a discrepancy between the definition and implementation of standards and new technology. Just now the packet TM/TC standards of CCSDS are coming into wide spread used in today's spacecraft. This is about ten years after definition of the standard. And even today they are not fully understood as the implementations show.

On the other hand, within CCSDS there are discussions on an interplanetary internet, which will lead to quite different features of satellite operations. Is it therefore worth it to stay with the „old TM/TC standards" and add some missing items?

Conclusion

The standardization of SLE services is a step forward in standardizing cross support. It will not solve all problems. But if first implementations are done and the software is available we hope more and more users will follow in implementing the services.

On this way some other important services have to be defined and implemented by the agencies to guarantee cross support services. The main items are:

- Tracking Services, which include antenna pointing as well as Ranging, Doppler and Angle services
- Time Correlation

References

[R1] *An introduction to CCSDS*, CCSDS A10.1-Y-2, May 997

[R2] *Packet Telemetry Standard*, ESA PSS-04-106. Issue 1. January 1988

[R3] *Packet Telemetry Services*. Recommendation for Space Data System Standards, CCSDS 103.0-B-1. Blue Book. Issue 1. May 1996.

[R4] *Cross Support Reference Model - Part 1: Space Link Extension Services*, CCSDS 910.4-B-1. Blue Book. Issue 1. May 1996

[R5] *Transfer Frame Processing*, Validation and Telemetry Packet Extraction, Functional Requirements Specification, DLR, Martin Fuchs, Issue 1.0, 16.6.97.

[R6] *Data Interface Specification*, Project DATANET, Volume III, Telemetry Subsystem, Revision F, DLR, Doc. Nr.: 868-SYS-230 (III), 7/86.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp.379-383

ISSN 0100-7386
Printed in Brazil

Reliability Testing or Reliable Testing

Curtis Fatig
NASA/GSFC/Orbital Sciences Corporation
GSFC/440.8/T-12F, Greenbelt, MD/USA
curtis.fatig@gsfc.nasa.gov

Sofia Stachel
NASA/GSFC/Orbital Sciences Corporation
GSFC/440.8/T-12F, Greenbelt, MD/USA
sstachel@hst.nasa.gov

Barbara Pfarr
NASA/GSFC
GSFC/584.0/Building 23, Greenbelt, MD/USA
barbara.pfarr@gsfc.nasa.gov

Abstract

Many new and refurbished control center systems are tested for system reliability, but once accessed by the end users the system is often labeled as unreliable. The testing of spacecraft operational control center systems has followed many different models. Often the basic ground system testing structure can be divided into three areas: unit or developer testing; system or functional testing; and operational testing, which occurs after the software has been delivered. Each of these areas focused on a different type and structure of the testing.

The authors have experience testing systems for NASA's Hubble Space Telescope (HST). Specifically, they have tested the newly redesigned Vision 2000 Control Center Systems. For HST servicing missions, they have experience assembling remote control centers used for integration and test of payloads. Their experience involved mission operations of the HST Orbital System Test (HOST) payload on Space Transportation System (STS)-95, an orbit test of equipment that will eventually be installed on the Hubble.

The original control center system was built in the late 70's and is expensive to maintain and upgrade. The need to reduce manpower and maintenance costs has precipitated the change. The STS-95 HOST remote ground system was a system built in the early 80's for hardware Integration and Test (I&T). The HOST control center used the I&T system as its core, with peripherals added to make it compatible with the communication networks and archiving systems used today.

The new Control Center system is a multi-server (server/client) system that incorporates commercial and custom software using 90's software and hardware.

The paper will define a process to assure the control center systems perform consistently, reliably, and meet the end users needs for a reliable real-time satellite control center.

Introduction

Many attempts at testing new and refurbished control center systems often have fallen short of the goal of providing a reliable system to the end user. With the increased use of Commercial Off-The-Shelf (COTS) software and hardware in real-time spacecraft operation control centers, providing a reliable system to the end user is becoming more difficult. Most of these COTS products are not intended to be exercised 24 hours a day, 365 days a year. Also, since the spacecraft operation control centers purchase a small number copies of the software and hardware, many manufacturers of COTS products are unwilling to invest in patch releases that occur out of their normal development cycle. In order to provide a reliable system to the end user, choosing these COTS products has to be integrated into the testing program. Additionally preventive maintenance has to be incorporated into spacecraft operations. The end users need to be actively involved in both these activities.

Approach

Previous testing of these complex HST systems has involved building similar ground systems at remote locations for unit and system testing with operational testing occurring at the operations facility. While this structure minimizes the impact to operations, frequently ground system problems are being found too late in the development and testing cycle to be fixed prior to being delivered to the end user.

In order to develop a reliable testing program that meets the end users needs, the following activities were developed:

- Reorganization of the test teams into a single coherent team involving developer, testers, and end users to resolve, fix, and test a majority of the problems found.

- Test the system in the environment it will be used in. Even if this creates additional up front cost, the long-term costs will be less and the end user will benefit greatly from the additional hands-on experience.
- If COTS products are to be used, purchase products the end user can work with operationally, instead of having the end user adjust critical operations to fit the COTS product.
- Base spacecraft ground system reliability success on real use of the system, not on projected and empirical data or testing results obtained in a clinical testing environment.
- Educate the development and operational managers on the importance of testing the system for reliability and providing the end user with a reliable system.
- Most critical to the success of the whole effort is to assign the persons actually doing operations to testing and development. The failure of many test programs is not having access to the right persons.

The HST program continues to pursue providing the end users with spacecraft operational systems that are reliable, effective, and within the realities of budget and manpower. These systems will be used for HST daily operations, as well as for the Third Servicing Mission in 2000, the Fourth Servicing Mission in 2004, and HST retrieval operations. The knowledge gained through the HST testing program can be applied to many other test efforts both commercial and government, to provide reliable systems meeting the end users needs.

Reorganization of the Test Teams into a Single Coherent Team

A single coherent test team involving developers, testers, and end users can be formed to resolve, fix, and test a majority of the problems found.

The advantages of a test team containing all personnel with a vested interest in the control center system outweigh the inconvenience that developers and users will feel. This coherent team will provide a better end product. Using this integrated philosophy will yield a reliable system that meets the expectations of the end users. Testers that have enhanced regression testing procedures, and developers will gain insight into the total system and not just their part of the code. As depicted in Fig. 1, the various phases of development and testing are divided into segments with the different disciplines of the team taking a lead or support role. In this way the team does not favor one discipline over any of the others. The figure also provides an example of the timeframe needed to accomplish the testing. This of course is dependent on many factors: number of changes to the system, availability of the system, problems found during development and testing, and the duration of the individual tests.

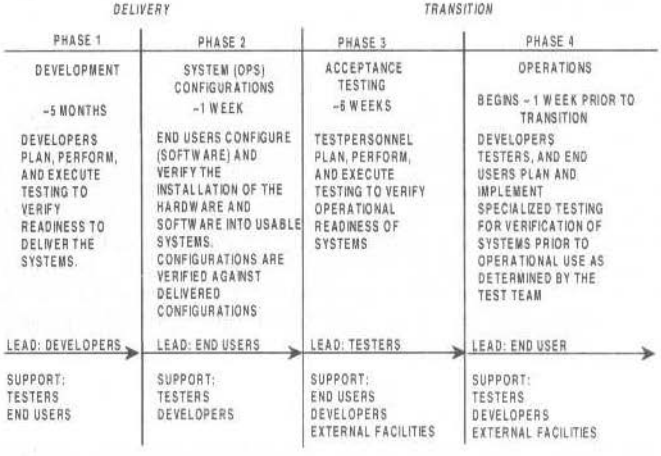


Fig. 1 Testing Phases

As an example, most of the releases of software of Hubble's new Control Center System were tested using the conventional method of system tests, followed by operational tools. The result of this testing method was a system that proved to be unacceptable to end users due to its poor performance in the operations environment. Before the next release became available for testing, the test team was completely reorganized to include not only testers, but also personnel from operations and development. This method is still in use, giving the end users great insight into their new ground system while

developers are gaining new knowledge on how their code can be further improved to serve the users better.

Assigning the right persons, actual end users, tester, and developers working on the systems may slow down the development and testing effort, but the gains will far outweigh the delays. Often the personnel available at the start of the test program contribute to the effort an incomplete set of skills for the task. Then when decisions need to be made about changes or implementations, the people making these choices are not fully qualified to do so. To the end users this means the system will be unreliable because of critical testing areas were missed due to lack of knowledge of the system being tested and of the operational environment in which it will reside. The drawback is a slower testing phase with test team members doing testing, as well as their current tasks. The benefit will be a system the users have faith in, because it has been more thoroughly tested, and hence will be a more reliable system.

Test the System in the Environment IT Will be Used in.

By testing the system in the environment it will be used in, even if this creates additional up front costs, the long-term costs will be lower and there will be added value to the end user.

The HST Vision 2000 is a client/server architecture with the users' systems based on Windows NT workstations. The client and server workstations and software, while reliable for some uses, under the requirement to operate 24 hours a day/365 days a year operations, have shown to be less reliable than the 'old' custom software system they are replacing. This has posed a unique problem of defining the type and duration of testing needed to assure a reliable system. The previous HST Vision 2000 releases met the requirement of 99.98% system up time when tested over a two week period. However, one month later, a process within the system 'crashes' every other day. A review of the code and equipment reveals no apparent cause for the system instability. Yet, if the system is completely re-initialized or rebooted all the problems go away for the short term, as in the previous two-week test period.

Using the target operational system for testing, instead of a clone or similar system in a test facility, will flush out the total system problems earlier in the program, which will reduce the integration time and create a system that will be more reliable for the end user. In the testing models used in the past, items such as user accounts, file access, privileges, and connectivity often were debugged late in the process which gave the end user the perception of poor configuration management and unreliability. With a single coherent test team these items can be corrected very early in the system development. In Figure 2, the testing levels and types of testing which will be exercised on the operational system are illustrated. The testing levels and types follow the traditional testing methods, but with the single coherent team on the operational system exercising these tests, the end users will have a reliable system to work with.

Preventive maintenance should be built into the development of the system and carried through to the operational system. Preventive maintenance includes: defragmentation and cleaning off files on hard drives, cleaning tape heads, exercising memory (virtual memory), and data throughput tests. Regular maintenance of systems will prevent some of the common problems which reduce system reliability and performance such as application speed, file volume storage, file transfer speed, and overall down time.

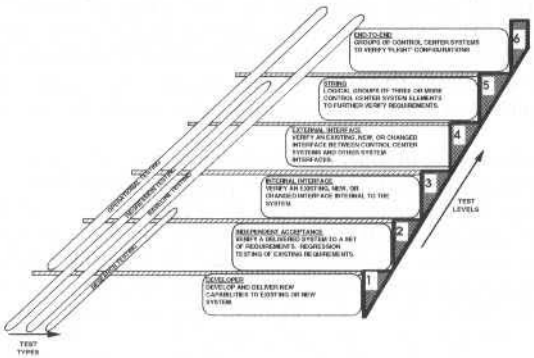


Fig. 2 Testing Levels and Types

COTS Products the end user can Work Operationally

If COTS products are to be used, purchase products the end user can use operationally; the end user should not be forced to adjust critical operations to fit the COTS product.

User input into the selection of COTS software is one area overlooked during the initial development of the control center systems that can increase the reliability of a system. As stated before, sometimes

during the initial system development, the users who are the ones best qualified to make decisions on whether an application fulfills operational needs, are not part of the COTS screening process. Instead, often it is the developers who evaluate several software packages, ability to meet the system requirements and sometimes the needs of the developer weigh much to heavily in the decision process. However, the letter of the requirement does not always fully convey the needs of the operations environment. The end user is the one who knows all the nuances of daily operations that, along with the system requirements, provide the complete description of a reliable ground system. Developers alone, or with consultant-type input from the end-user, cannot choose the best COTS for a new system. Without the end user deeply involved in this selection, many times the COTS software functions are within the confines of the system requirements but fall short due to unforeseen human factors.

An example of this occurred on Hubble's Vision 2000 system. The development team, with limited input from the end users, chose the Netscape browser as the end user interface for all functions. Netscape performed well in the development environment and met all requirements. When the end users started using the Netscape browser in operational testing, before the integrated test team approach was adopted, problems were encountered, not only with virtual memory errors, but also with the end users' preconceived knowledge of the browser. The Netscape interface did not match all the needs of the control center system interfaces and attempts to adapt the users to compensate for poor performance failed. If the developer and the end user had worked together in the COTS selection, some of these problems could have been found and resolved.

System Reliability Success Based on Real Use of the System

Spacecraft ground system reliability success should be based on real use of the system, not on projected and empirical data or testing results obtained in a clinical test environment.

The control center system to be used for mission support should also be the system used for spacecraft development. Of course during development, the control center system will contain tools and functions needed for spacecraft development that will not be used for mission support. This also means that the developers should use operational procedures and scripts during the development phase. This will reduce the number of procedure errors during mission operations. The development system also should ingest real spacecraft telemetry in an environment that is more like mission operations than a normal development facility. The appropriate subsystem engineers should analyze this data for completeness and accuracy during the earliest stages of development.

This use of the control center system during the spacecraft development and testing will provide system reliability data before mission operations begins, using data from the real spacecraft.

Educate the Development and Operational Managers

The development and operational managers need to be educated on the importance of testing the system for reliability and providing the end user with a reliable system.

Many times, schedules, high visibility testing, and workarounds to 'make' the system work take precedence over the testing of system reliability. As was stated in earlier sections, with the single coherent test team personnel participating in the development, testing and integration of the new system a slower pace of activities is likely to occur. However, the benefits of creating a more reliable system, with fewer post software/hardware release fixes, and greater end user satisfaction far outweigh the delays in schedules. This point is often lost when management focuses on the product delivery dates and not the quality of the product.

In an effort to deliver the system in time, or to have the system ready for a milestone test, the system is hastily patched to get through these situations. This patch is intended to be a temporary measure, but too frequently a temporary fix becomes a permanent solution. But treating a symptom does not necessarily treat the illness, and a potentially serious problem can be hidden by the workaround until the system's reliability begins to suffer due to the existence of so many patches. Therefore it is very important to work with management throughout the development of the system so that management understands the risk involved in forcing the development cycle to adhere to inflexible schedules. By the same token, development managers must present realistic delivery schedules that allow their team to create and fully test the system. If delays or setbacks occur, both management teams must work together to create a new schedule that ensures enough time for proper development and testing but also supports the end-users' priorities.

Reliability Testing and Real-Time Operations

Once the system is delivered, the testing structure that validated the system continues to have a pivotal role. This team can assist the end users in refining and improving their procedures to take full

advantage of the new system's capabilities. They should continue to monitor system performance and reliability to ensure long-term success. They can also work with the users to come up with system enhancements and capabilities to the ground system made possible by new developments in technology.

Reliability Testing

Once a system is delivered, all the end users are involved in the system. Now is the time for the testers and developers to evaluate the system to see if pre-delivery performance models and assumptions are correct. In a recent delivery of the HST control center system, the team approach discussed in this paper was used very late in the process. The results, even at the late stage in development, had many positive effects on the reliability of the system. These positive effects were:

- Reduced end user resistance to a new system
- More testing, particularly during off hours
- End user investment in a reliable working system
- Sharing information between end users, testers, and developer on the good and bad points of the new system.

With the interaction between the various teams during the first few weeks of systems use, the general impression of the system was very positive. The end users' problems with the system were given more attention than in the previous system releases. This investment of the end users had the benefit of increasing their self worth in the whole process. With the end users' positive approach to the process, the system reliability increased due to a combination of improved system perception and quick investigation and resolution of problems.

Real-Time Operations

In real-time operations, the new control center system is being used in a less stressful way than during the testing phase of the system. With the end users now operating the system full time, system stability and reliability are increasing, as the end users become more familiar with the system. As with any system, the more you operate it, the more you become familiar with it, the fewer problems you generally have with the system. The lesson to be learned is to involve the end user in the system development as early as possible.

Summary

No single event will increase system reliability greatly, but when the above items are taken as a whole the system reliability for the end user dramatically increases.

Reliable control center systems, with minimal end user training, will be in great demand in the International Space Station era, when many control center systems are moving to a client/server architecture. The client/server architecture has many advantages, but with this dispersed architecture, system reliability is more difficult to achieve. One way to achieve a more reliable system is to incorporate technologies that the developer and end users' agree upon. There is no reason why a reliable control center system cannot be built today, using currently available software and hardware, if the above guidelines are followed.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp.384-389

ISSN 0100-7386
Printed in Brazil

Operability Testbeds for the Next Generation Space Telescope

Keith Walyus
NASA
Goddard Space Flight Center
Code 581
Greenbelt, Md. 20771 USA
kwalyus@hst.nasa.gov

Glenn Cammarata
NASA
Goddard Space Flight Center
Code 582
Greenbelt, Md. 20771 USA
glenn.cammarata@gsfc.nasa.gov

Roger Rowe III
Altair Aerospace Corp
4201 Northview Dr
Suite 410
Bowie, Md. 20111 USA
rrowe@altaira.com

Chris Wheal
Altair Aerspace Corp
4201 Northview Dr
Suite 410
Bowie Md. 20111 USA
cwheal@altaira.com

Abstract

The Next Generation Space Telescope is scheduled to launch in 2007. It will operate primarily in the infrared spectrum and achieve a high resolving power by taking advantage of advances in lightweight optics and electronics. New instrument and operational technologies are being examined to accomplish the mission's lofty scientific goals. Two operational technologies that are being examined are "event-driven scheduling" and "finite state-modeling". For an event-driven system, the commands would execute based on events, potentially decreasing the time the spacecraft waits for its next command to execute. Preliminary calculations have determined that up to 3% more observations could be accomplished with an event-driven system versus the conventional time-driven system. Advanced on-board modeling techniques will be used to more efficiently display the spacecraft's state. Additionally, this same state information could also be input into the event-driven scheduling system, as the scheduling system will need to assess the spacecraft's state before undertaking a new activity.
Keywords: *Testbed, Scheduling, State Modeling*

Introduction

The Next Generation Space Telescope (NGST) is a key component in NASA's Origins Program. Scheduled to launch in 2007, NGST will be a large, passively cooled telescope, which will build upon the knowledge already gained from the Hubble Space Telescope (HST). Science priorities for NGST require the following capabilities:

- Sensitivity superior to HST and ground-based telescopes to detect very faint stars and galaxies,
- Angular resolution comparable to HST to clearly separate two near-by targets and avoid confusion and overlapping images,
- Wide field of view comparable or larger than HST to measure many objects at one time for surveys, and
- Optimized performance in the near infrared portion of the spectrum to observe distant galaxies and stars and to survey heavily obscured regions of star and planet formation.[1]

The technology for the NGST instruments will far surpass the instruments launched originally on HST in 1990, and even those that will be launched on the final HST servicing mission in 2002. But just as instrument technology has steadily evolved since 1990, so too has the operational technology and practices. Although new instrument technologies will be necessary to accomplish the ambitious science agenda, the NGST should be a relatively simple system to operate, and no new operational technologies have been identified as being essential to conduct the mission. Still operations engineers have sought to identify technologies that will increase mission safety or operational efficiency.

Because the launch of NGST is still over eight years away, the final design and operational concept for NGST have still not been finalized, although preliminary concepts have been studied. In the subsequent years, engineers hope to test various technologies, which may later be applied. Various methodologies will be used to develop and prove these technologies. Clearly all studies start with the initial research and the development of ground based models. These models can be executed on personal computers or workstations, or as a further step, executed on ground-based simulators. As a final step, some technologies need to be proven on existing satellites before they will be accepted for use for new missions. To this end, NGST engineers have identified candidate missions that could be used for this purpose.

Two missions that have been identified as potential candidate missions are NASA's Wide Angle Infrared Explorer Satellite (WIRE), which is scheduled to launch in February 1999, and the Unites States Air Force's (USAF) Mightysat 2.2 satellite, which is scheduled to launch in June of 2002. The WIRE spacecraft is the fifth in the series of NASA's Small Explorer spacecraft. Its primary mission is to conduct a survey in the infrared bands, to better understand the formation and evolution of galaxies. Because it uses a cryogen dewar to cools its optics, its primary mission length is limited to only 4-6 months. After this time, the science instrument will no longer be able to provide meaningful data, but the satellite will still be perfectly healthy with an expected mission lifetime of several years. NASA engineers are developing a testbed program for the WIRE satellite, which will commence upon the completion of the primary science mission. The testbed program is currently planned to last for one year, although it could be extended based on the number of proposals submitted and the funding available. Current funding estimates are for two full-time equivalent (FTE) personnel per year. One FTE will be for Flight Operation Team (FOT) support to operate the spacecraft. Three-quarters of another FTE will be needed for flight software support, while the remaining quarter of an FTE will be budgeted for program management. Each proposal team will provide their own personnel for data analysis.

The primary goal of the program is to operationally test new tools and algorithms to better quantify their operational effectiveness. Although many capabilities can be functionally proven through ground tests, operational experience is invaluable for quantifying the effects on the overall operational concept. As a secondary goal, the program will be used to train new FOT members. Various proposals have already been submitted for this testbed program. Proposals include tests of new star identification algorithms, advanced communication protocols, new approaches for spacecraft attitude and orbit estimation, and an on-orbit demonstration of Finite State Modeling.

The USAF Mightysat satellites are a series of satellites devoted to technology demonstrations. The goal of the Mightysat program is to provide frequent, low-cost access to space for the demonstration of emerging technologies. The USAF's Phillips Lab manages this program, which is located at Kirkland Air Force Base in Albuquerque, New Mexico. Mightysat 2.1 is scheduled to launch in June of 2000, with 2.2 scheduled to launch two years later. Various proposals are still being examined for incorporation onto the 2.2 satellite. Event-based scheduling and Finite State Modeling techniques are two proposals being evaluated for this satellite. Results from the earlier WIRE mission will be used as an input to a more ambitious test experiment to be flown on Mightysat.

Finite State Modeling with the *Altairis* Mission Control System (*Altairis*)

One of the primary functions of any FOT is to monitor the health and safety of the spacecraft. Traditionally, the FOT would monitor the numeric telemetry for a discrete set of parameters. The data would typically be represented as numeric values or possibly be displayed by simple plots. Teams might develop procedures to assist in this very labor-intensive task, although the development of these procedures would also be very labor-intensive. Frequently these procedures would not be well coordinated and would more likely reflect the views of an individual or a group of individuals from the FOT. As operational technology has progressed, teams have taken to develop rule-based and case-based systems. These systems have proved very effective for small applications, but become unwieldy for larger, more complex missions.

As an alternative system, the Altair Corporation has developed a monitoring system based on Finite State Modeling. Unlike conventional health and safety monitoring, Finite State Modeling does not rely on monitoring individual telemetry points, but rather combines telemetry points to represent various spacecraft states. By representing the system as a state, the current telemetry can be compared to pre-defined states to ascertain the spacecraft's status.

In the representation of complex systems it is useful to consider two mappings from the real world of components. These two mappings are the Physical Model and the Functional Model.

Physical Model

The physical model follows the actual physical layout of the vehicle and includes every component of interest. These components are modeled in an object-oriented database with their *attributes* described in sufficient detail to provide a complete physical description. Each component is assigned to a class or sub-class through which it inherits some or all of its attributes. These may be physical attributes such as its name, part number, serial number, batch number, location, manufacturer, date of manufacture, version number and operating limits, or they may be functional attributes such as battery charge characteristics, tape recorder duty cycle limits, associated telemetry sensors, low level commands and procedures.

The object attributes will also include *methods* associated with the individual component or data object, including associated statistical operations, engineering unit (EU) conversion coefficients, and other user-defined characteristics. The physical model thus provides the mechanism for tying commands, data, calibration coefficients and data variables directly to components, relieving operators of the need to remember (or look up) telemetry mnemonics as any data they need can be viewed with the Altairis Model Explorer.

The physical model connects data into logical component groups, providing the basis for the data distribution model. This also allows downlinked data to be labeled with satellite and orbital plane IDs as well as information about a particular component to be routed to the correct location for processing.

Functional Model

The functional model overlies the physical model and comprises the same elements but is arranged by system and subsystem rather than by physical location. The same component may be shared by more than one system and thus may appear in more than one place in the functional model. An illustration of the mapping of components into physical and functional models is shown in Figure 1. The functional model represents the manner in which the individual components defined in the physical model interact with each other to form working systems and subsystems. It is where the operational knowledge of the vehicle engineers is captured and where the potential for automation lies.

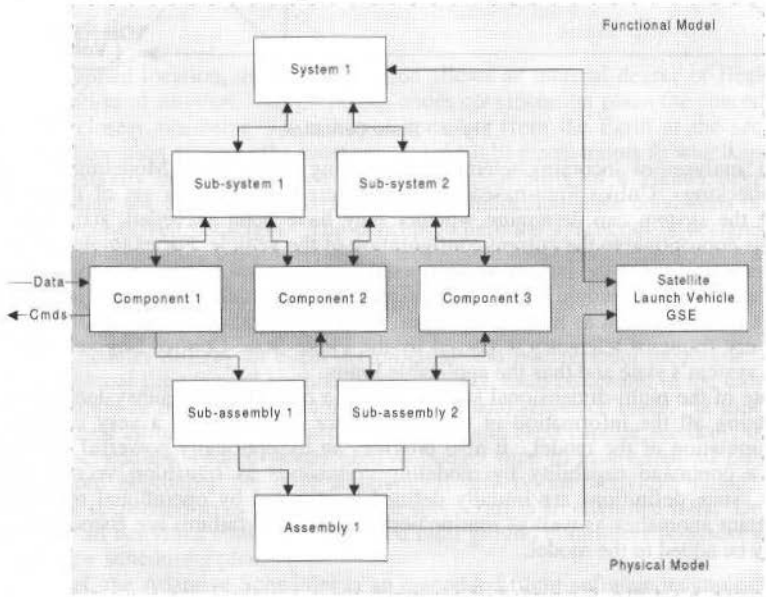


Fig. 1 Physical and Functional Models

The system to be modeled is represented by a hierarchy of vectors in state space, each representing a subsystem of the one above. At the lowest level components are also regarded as systems. The dimension of the state vector of a system is equal to the number of data items in its definition. For example, the state vector of a relay could take one of two values corresponding to its bi-level telemetry signature and it is thus a one-dimensional vector. The state vector of a relay control unit consisting of several relays is defined by a group of telemetry signatures indicating the states of its constituent relays, each of which is a component, or a dimension, of the state vector. Figure 2 shows an example of a three-

dimensional state definition, in this case a battery whose state is defined by its voltage, current and temperature.

At a higher level the state vector of, for example, the electrical power system (EPS) is defined by the states of its various constituent parts such as relay control units, batteries, charge control units, solar arrays, etc, each of which is a component of the EPS state vector. At still higher levels, the state vector of the entire vehicle is defined by the states of all its systems and the state of a constellation of satellites is defined by the states of each spacecraft in the constellation. At the highest level, the state of a satellite communication system is defined by the state of the constellation and the state of all the ground stations.

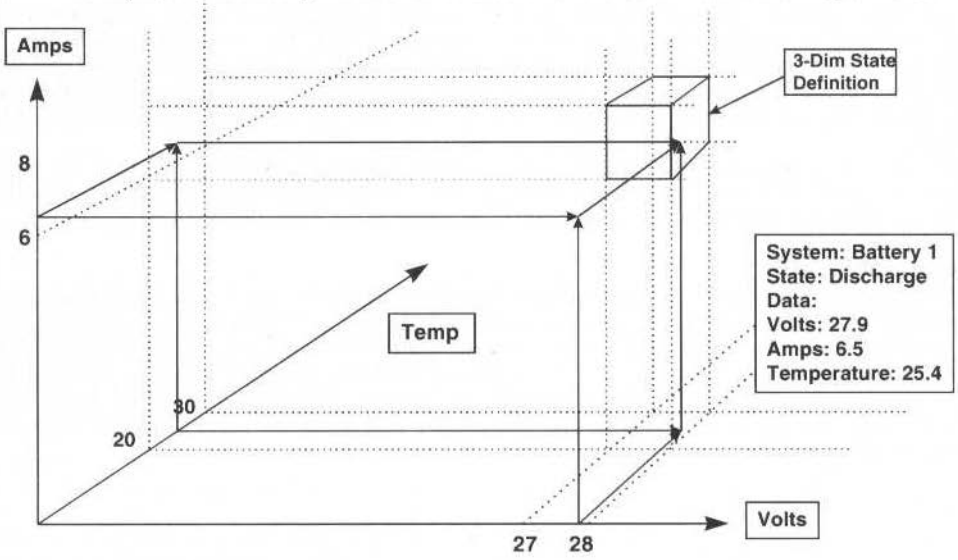


Fig. 2 State Definition

Automated analyses of incoming telemetry data using Finite State Modeling is much more than merely limit checking. Unlike limit-based systems, where the particular set of limits in use must be defined before the system can determine whether they have been exceeded, states are defined using functional limits appropriate to the current configuration of the vehicle. For example, when the spacecraft is emerging from orbit night the battery voltage would be expected to be lower than at the end of orbit day and the state definition would reflect the difference. State definitions may include statistical and time-dependent functions such as moving averages, standard deviation, or pattern matching for transient signatures. Every frame of telemetry is parsed by the Finite State Control Engine, which continuously determines the system's state and thus the applicable limits.

The collapse of the multi-dimensional state vector of a component or subsystem into a single named state encapsulating all the information in the state vector itself, allows a very compact notation and simplifies manipulation of the model. It also provides an exceptionally powerful and compact way of implementing a command capability by modeling commands as transition vectors between defined vehicle states. State definitions are initially defined as dictated by operational relevance and include likely or important anomalies as well as routine normal states. If failures are encountered in operations, these can easily be added to the model.

Flight Software and On-Board Autonomy

Because the hardware and software architectures of ***Altairis*** are fully distributed, any ***Altairis*** process or object can be distributed to any node in the network, including processors on board the satellite. Thus ***Altairis*** has the potential for arbitrarily large migration of functionality from ground to space. A suggested architecture would host a Finite State Control Engine (FSCE) on each satellite running under a real time operating system and will include a statistical data analysis and a trending analysis process.

An essential feature of this design is that it will be a *single* integrated system, not a ground system interfaced with individual space systems for each satellite. Only in this way can the entire system exploit the advantages of the model structure and the power of object oriented software. The concept of a fully integrated system means that the traditional distinction between ground and flight software is no longer necessary or relevant. There is now only a single class of software and it is consistent throughout the

system, within the constraints posed by the requirement that any ***Altairis*** processes hosted by the on-board computer will necessarily be running under a different (i.e. real time) operating system. This will not, however, affect the functionality of the on-board processes.

During initial operations of a satellite, the ground segment must retain control for a period after launch until enough confidence is gained in the control system to allow the satellite to operate autonomously. During this period satellite systems will be monitored continuously using high rate telemetry and the output of the onboard ***Altairis*** processes compared with that of the ground based master system. As confidence is gained, control will be transferred progressively from ground to space until fully automatic routine operations are achieved. The use of ***Altairis*** as the single, integrated ground and flight control system will greatly facilitate this migration. All software processes, finite state models, transitions and procedures, being common to both the space and ground segments, will remain unchanged during and after the transition.

When the system has fully migrated, the onboard FSCE will run the full Finite State Model of the satellite and will conduct its own state analysis and short and long term health monitoring and trending. During routine operations apart from scheduled maintenance the satellite will transmit only top level vehicle and system state information to the control center. This reduction in orbit-to-ground data transmission will have obvious beneficial effects on the overall bandwidth requirements.

Adaptive Model

The adaptive model will utilize the output from the statistics and trending processes to refine the boundaries in state definitions in a way that will improve the model's predictive powers and assist in long term health monitoring and trending. The statistics and trending processes will generate histories of parameters, including such statistics as average, standard deviation, average time rate of change, etc. The Adaptive model will adjust its boundaries based on the trend data and a set of criteria defined in the model. In this way, the model becomes less rigid, allowing for the natural degradation of the spacecraft over time without loosing the ability to recognize anomalous behavior.

Adaptive Scheduler

Because of its planned location, the NGST mission allows an unusual degree of flexibility vis-à-vis scheduling and execution of mission. All proposals under consideration place the spacecraft well away from the Earth, the closest site being 1.5 million kilometers from the Earth at the second Earth/Sun Lagrange point. This location removes the most severe scheduling constraints to which spacecraft in low Earth orbits are commonly subjected, which suggests the feasibility of a greater degree of onboard autonomy. In particular, we want to take advantage of the immediate state conditions of the spacecraft and mission in an event-driven paradigm. The concept, where the spacecraft is designed with a fairly autonomous short-term scheduling capability, which will enable itself to execute science and engineering activities, has come to be called an Adaptive Scheduler. The purpose of the Adaptive Scheduler is to reduce normal operational costs by increasing on-board autonomy, making more efficient use of on-orbit time, and simplifying the ground system processing.

The Adaptive Scheduler utilizes an event-driven command management paradigm. This approach differs from the traditional time-tagged method. Traditionally, each command in the onboard command sequence has an absolute or relative time stamp. If a command sequence reads, "at time t+20 slew the spacecraft to the desired attitude, t+50 acquire guide stars, and t+60 capture science", then the ground based planning and scheduling system would need to budget time to insure that all previous activities have completed before beginning the next activity. If any of the activities failed, the remaining observation would be suspended so as not to leave the spacecraft in an unknown state. Depending on the severity of the error, all activities could be suspended until ground controllers re-assess the spacecraft's status and uplink a new scheduling plan.

At a concept level, the Adaptive Scheduler is an embedded flight software technology that manages the execution of the science observation program. This includes but is not limited to the management and synchronization of attitude maneuvers, guide star acquisitions, science instrument configurations, science data collections, and routine spacecraft health and safety activities. For an adaptive scheduler, activities are not rigidly controlled by time, but rather are driven by events. Events include low-level spacecraft activities (e.g., slew to target, acquire guide stars, and capture science). A time-driven event could even be defined as an activity, much like a traditional scheduling system, although the use of time-scheduled events are minimized so as not to restrict the system's flexibility. If a failure occurs, the Adaptive Scheduler would then evaluate the status of the spacecraft, possibly using a system like Altair's Finite State Modeling System, and then re-schedule its activities to maximize observing time.

Figure 4 shows the Adaptive Scheduler receiving an observation list from the ground. An observation is a high level construction consisting of lower level spacecraft activities. Each activity has two parts, pre-validation and execution. All activity pre-validation within an observation must complete successfully before any activity execution is initiated. In this example, the observation list contains at least four observations. In an event-driven manner, the scheduler completes observation 1, autonomously inserts a routine engineering activity (momentum dump), completes observation 2, bypasses observation 3 because of a pre-validation failure, and finally completes observation 4.

By employing an adaptive scheduling system, additional observations could then be conducted over a finite time period. Studies have shown that such a scheduling system could increase the number of science observations by 3% over the life of a mission. Additionally, operational costs would be reduced as more functionality could be transferred to the satellite.

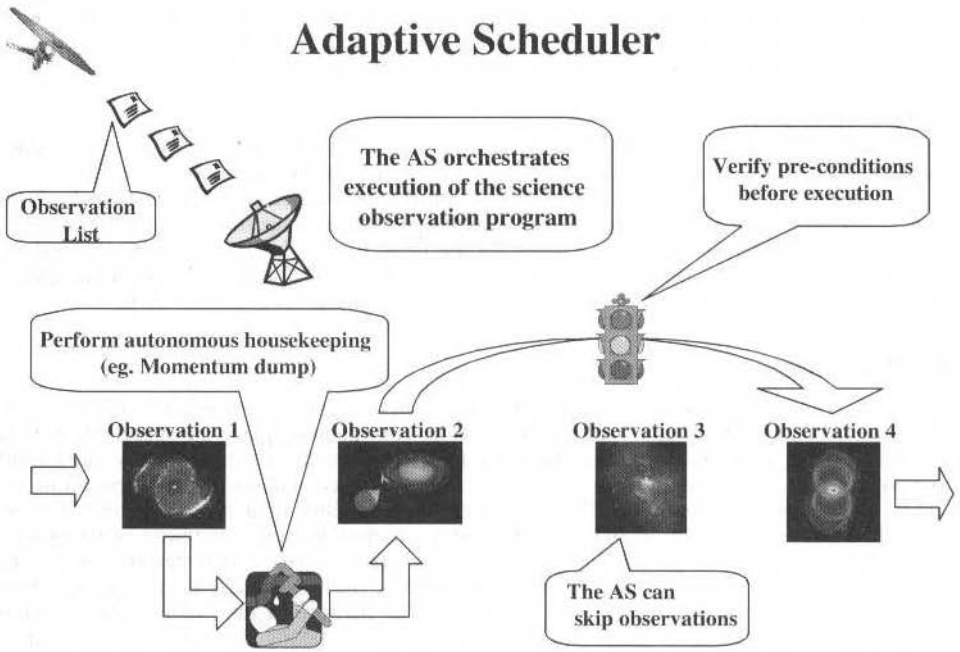


Fig. 3 Adaptive Scheduler Concept

Summary

GSFC engineers are actively pursuing various ways to operationally test promising new technologies. By testing these new ideas on testbed satellites, a greater understanding of the overall operational effectiveness will be understood and their effect on the missions operations concept. The two technologies listed in this paper are just two of the technologies being assessed for NGST, yet NGST will not be the only beneficiary. Finite State Modeling is currently being applied to the SOHO ground system and will be used by the Earth Observing System (EOS) AM-1 satellite's ground system. Adaptive scheduling technology is also being examined for other GSFC satellites. By using technology-demonstration satellites and satellites which have exceeded their prime mission for validating new concepts, GSFC engineers expect to reduce mission risk when these technologies are introduced into future, more ambitious science missions.

References

[1] *Next Generation Space Telescope, Visiting a Time When Galaxies Were Young*, ed. H.S. Stockman (The Association of Universities for Research in Astronomy Inc.,1997), p. 16.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp.390-396

ISSN 0100-7386
Printed in Brazil

Spacecraft Attitude Anomaly Resolution as an Example of the Application of Expert Technology in a Quasi-Autonomous Operations Environment

A.J. Bradley
Allied Signal Aerospace Guidance & Control
20701, P.O. Box 91, Annapolis Junction, Maryland, U.S.A.

S. Sobieski
Swales Aerospace
20705, Beltsville, Maryland, U.S.A.

J. Leibee
NASA Goddard Space Flight Center
20771, Greenbelt, Maryland, U.S.A.

Abstract

This paper describes a prototype system, which addresses the attitude determination problem space in the Hubble SpaceTelescope Pointing Control System domain. It is being developed as part of the reengineering of the ground system, an effort undertaken to substantially reduce operations cost. A new quasi-autonomous architecture has been developed for this effort. It is considered quasi-autonomous in the sense that human operators are only required for collaboration with the Expert Systems to resolve new problems and, once resolved, for directly updating the Knowledge Base supporting the Expert Systems. The prototype is employed within the System Monitoring and Anomaly Resolution segment of the architecture. It uses multi-paradigm reasoning, consisting of production Rules, Cases and Neural Nets. Future enhancements will include a Truth Maintenance System, Bayesian Nets and models of some spacecraft elements. Four classes of commonly encountered anomalies have been defined, which per force *our system must handle. They include failed events, unplanned events, subsystem anomalies, events accomplished in an unusual manner, abnormal trends or unusual correlation with the orbital environment.*

Keywords: Spacecraft Ground System Reengineering, Quasi-Autonomous Architecture, Attitude Determination Automation, Multi-Paradigm Reasoning.

Introduction

One of the primary goals of the Hubble Space Telescope (HST) reengineering initiative, entitled HST Vision 2000, is to "substantially reduce the cost of operating and maintaining HST". This paper presents a concept developed for Vision 2000 which reduces staffing levels by using a transactional architecture that relies on Expert Systems for performing much of the spacecraft sustaining engineering and anomaly resolution. It operates in a quasi-autonomous mode. Autonomy may be defined as the capability of a system to accomplish a prescribed goal by selecting and accessing appropriate resources or data, proceeding through an efficient sequence of functions, and taking any remedial actions necessary for successful task execution - all without human intervention or direction.

We define a quasi-autonomous system (QAS) as one in which routine deterministic functions can be designed to be performed without human intervention, but those requiring judgmental analysis, such as those encountered in sustaining engineering and anomaly resolution, are accomplished by a collaboration between the operators and an Expert System. D. Fish discusses the criteria and rationale for selecting the appropriate level of autonomy within the reengineered HST system, (Fish, 1999). In a QAS, as new operational or system anomalies are encountered and resolved by the domain experts, the remedial procedures are then added to the system's Knowledge Base (KB). The system is thus continuously learning and grows more autonomous with time, guided by its human mentors. [It is important to note that this approach posits an operational expert database that can grow and is readily generalized for future mission application. When developed, it would provide a computer readable lessons learned lexicon which could minimize operational risks in succeeding generations of missions as well as reducing operational procedure development cost.]

The domain experts are the Mission Engineers who are primarily responsible for monitoring the performance of the spacecraft and responding to anomalies. As part of its sustaining engineering task, the Expert System monitors and analyzes the status of all designated spacecraft subsystems, prepares performance assessment reports, and, if non-nominal changes occur, recommends adjustments to the

reference data also being maintained in the KB. These data, along with appropriate contextual and historical case data, are required for diagnosing anomalies.

If cost savings are to be realized, a reduction in Mission Engineering staffing cannot be at the expense of an increase in the software or knowledge engineering staffing required to maintain the Expert System and, specifically, the KB. Its maintenance must be simple enough to be performed by the domain experts directly. This approach demands that the software tools employ a relatively easy user interface for knowledge definition and maintenance. The methodology, using the attitude control subsystem as case example, is discussed below.

Architectural Concept

Figure 1 shows a high level version of the transactional concept for operating a mission. Although it was developed for Vision 2000, this architecture is quite general and is applicable to a wide range of missions (or tasks). All processes except for the initial mission planning and the default anomaly resolution- shown as shaded steps in Fig. 1 - can proceed without human intervention. The allocation of the processes between ground and space for any mission depends upon mission complexity and availability of duplex communications. For HST, the operational system is largely ground based.

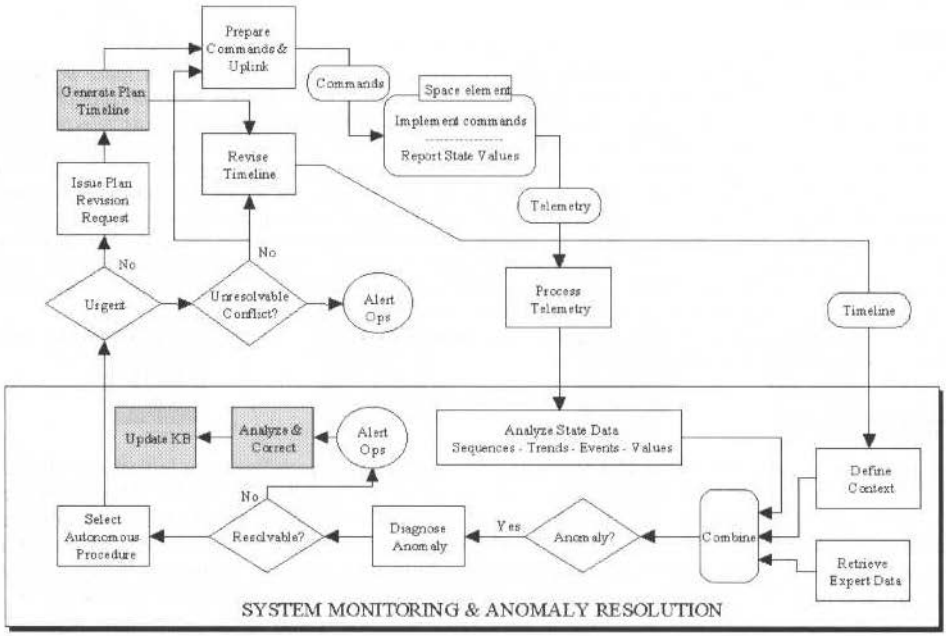


Fig. 1 Quasi-Autonomous Transaction Flow

The space element (HST) executes up-linked commands according to the timeline specified in the mission plan. It returns telemetry from various on-board sensors which is analyzed by the monitoring function – tasks enclosed by the shaded box in Fig. 1 - to determine the states of all designated components and to detect anomalies in the operation. Four classes of anomalies must be considered:

- Failed events and unplanned events including communication, configuration, attitude, and observational activities.
- Failed or problematic functioning of subsystems indicated by non-nominal telemetry values or values in anti-correlation with the environmental context.
- Event accomplishment but with unusual process metrics such as duration for accomplishment or unusual number of discrete steps required as compared to historical records.
- Subsystem metrics showing unacceptable trends or unusual correlation with the orbital environment, which can indicate wear, degradation or unusual response.

The context is derived from the predicted event timeline. It is used to provide the parameters needed for qualifying each monitored state value. Trend analysis records are updated as well. The system utilizes

a mixed paradigm of production rules, neural nets and Case Based Reasoning (CBR), in an object-oriented environment, to detect and analyze anomalies and subsequently selects a remedial command procedure (designated as an Autonomous Procedure– AutoProc) from its historical knowledge database in response. The AutoProc contains the data requirements, the processes or functions employed and rules to be invoked. In this way, only well-defined and validated remedial command procedures are employed to absolutely minimize risk to the space asset. Urgent problems lead to near real-time response or a response as soon as a communication link is active; those which are non-urgent result in a plan revision for future implementation. Unresolvable anomalies result in an alert to be issued to system engineers who, after resolving the anomaly, update the knowledge database, including criteria for detection and the respondent AutoProc. The interaction of all elements, sequencing or resequencing of tasks and general process oversight is provided by an autonomous Task Manager element (not shown in the diagram).

The complete Expert System would emulate all the manual tasks performed by the HST Pointing Control System (PCS) Flight Operations Controllers and System Engineers. These tasks range from real-time monitoring of spacecraft telemetry, to extracting and processing archival data for long-term trending, to diagnosing and responding to anomalies. Attitude determination and the resolution of anomalies encountered are an integral and critical element of this task. Each of these tasks is unique, they require separate and distinct knowledge, and the processes by which they are performed are unrelated. Multiple paradigms have therefore been employed, which include procedural rules, case based reasoning and neural nets, in an object oriented environment. The prototype system is limited to demonstrating an approach to automate the presently labor intensive HST *post facto* attitude determination task. Eventually it could be extended to an on-board instantiation to afford fully autonomous operation.

Domain Description

The HST PCS has previously been described in the literature, (Bradley, et al., 1991). Attitude determination can be conducted with the Star Trackers and with the focal plane Fine Guidance Sensors, our focus in this paper will be on the Star Trackers. There are three Star Trackers on HST; any two can be used for three-axis attitude determination. They are analog PMT instruments with an 8.5x8.5 degree field-of-view (FOV) and an Instantaneous-Field-Of-View (IFOV) of 10.5x10.5 arc minutes. The Star Trackers have three search modes, Search Scan, Offset Scan and Track Scan. Attitude corrections are performed on-board in real-time with the Star Trackers in Offset Scan mode, whereas attitude determination is computed post facto on the ground with the Star Trackers in Search Scan mode.

In Search Scan a Star Tracker traverses the entire FOV in approximately 80 vertical (V) steps from the –V to the +V edge, each step is accompanied by a horizontal scan. When a star is detected above a predefined threshold, the Star Tracker transitions to a Track Scan and remains locked on the star. A Break Track command causes the Star Tracker to step approximately 0.7 degrees in the +V direction (denoted as blanking) and resume searching for stars. Through this process the Star Tracker will incompletely map all stars in the FOV. The map is incomplete since blanking will result in some stars being missed. The telemetered results of Search Scan are used for post facto attitude determination on the ground.

Offset Scan mode is used for on-board attitude correction. The Star Tracker searches for a specific star in a 1.5x1.5 square degree Reduced-Field-Of-View (RFOV). Within the RFOV the Star Tracker steps in V and scans in H searching for the star. When a star is acquired, the Star Tracker transitions into Track Scan mode. If the acquired star does not meet the predefined criteria, then a Break Track command is issued and the Star Tracker resumes searching for the correct star.

An Approach to Automation

Automation can be implemented using a judicious collection of commercial and government software tools, with the goal of quick and simple implementation. The resulting system, however, requires user interdiction to apply meta-level knowledge and task management to make the various tools play-together to produce a useful product. Our approach attempts to integrate the disparate elements using network messaging, meta-level knowledge and an automated task manager to minimize user intervention, except in cases where a problem arises which is either not covered by the KB, or constitutes a spacecraft emergency.

The approach uses a Real-Time and Off-line Expert System (RTES, OLES) and incorporates multiple paradigms, in the belief that one single reasoning paradigm will not suffice because the reasoning the human experts perform on the diverse tasks are not the same. Some tasks may call for direct inferencing, while others can best be resolved by matching a known case, or signature pattern matching. Several multi-paradigm systems have been described in a recent Workshop,(Bichindaritz, et al., 1998, Chen, et

al, 1998, Oertel, et al., 1998), which incorporate CBR. In a departure from previous systems, CBR is not designated the master, or servant to other paradigms, rather it is integrated into a multi-modal reasoning architecture, where cases and rules separate subjective and objective knowledge. Other paradigms that are under consideration for the future are Truth Maintenance and Bayesian Nets. The knowledge acquisition process employed for this system facilitates ease of KB population to support the RTES and OLES elements.

The RTES receives hundreds of telemetry parameters at a one hertz rate; it is designed for rapid performance to detect limit violations, control system events and state changes. The real-time inference engine consists of production rule sets in a forward chaining implementation. Rules are separated by context, where context can indicate an orbital environment, a spacecraft configuration, or an event in progress. Context partitioning keeps unnecessary rules from firing. Anomalies detected are sent as messages to the OLES. The OLES combines the paradigms of production rules, neural nets and case based reasoning, in an object-oriented environment. Messages are received from the real-time system, or user interface, requesting information or anomaly analysis. The OLES is where attitude determination, long-term trending of spacecraft parameters and events, and anomaly resolution take place. While the RTES detects a finite set of immediate anomalies, trending can uncover long term conditions like degradation of hardware, or indicate an impending limit violation based on the derivative of a parameter trend. The OLES has access to the entire KB, which includes the as-executed command timeline, full telemetry archive, historical anomaly reports, environmental conditions, ground and flight reference databases.

The present development environment has the real-time system based upon RTworks products, which are resident on an SGI Indigo2. The off-line system, which is based upon LiveModel and PowerModel, is resident on a Sun Sparc20. The Neural Net tool and the CBR are resident on a PC in a Windows environment. The runtime environment will be solely PC based. The RTworks inference engine can be resident on a PC, the LiveModel and PowerModel applications are ported to a Windows Visual C/C++ environment.

Learned lessons from an earlier effort, (Siemens, et al., 1986) to develop an Expert System for satellite anomaly resolution have been factored into our approach. These covered rules partitioned by context, accounting for multiple manifestations of a single fault and problems due to the knowledge engineering process itself. Partitioning rules by context is one method to reduce the complexity of very large rule databases. Meta-rules are needed to detect multiple manifestations from a single fault, wherever the rule-based architecture permits a fault to effect more than one object. However, this is one area where not all combinations of multiple manifestations can be accounted for, meta-rules are only implemented for faults which the domain experts deem reasonable.

Knowledge Base Development

The domain experts for HST have the bounty of eight years of historical telemetry, numerous failure reports and years of simulations. The KB for HST has the potential to be defined to a level of depth not encountered before on previous spacecraft. While all this is beneficial, it does present a challenge in constructing a large, detailed KB and having it perform as well as, or better than, a human counterpart.

For the RTES, the KB is constructed of forward-chained production rules. The rules are entered via a menu driven user interface, which provides a uniform rule template, so that all rules tend to have the same structure. The first step in defining the KB is to establish the Class-Object-Attribute structure. Classes and objects are entered into the KB via a menu driven user interface.

For the OLES, the combination of Live Model and Power Model software tools are used to create an object-oriented environment for the user to define classes and objects, and to establish Event Diagrams that define the flow of information between the objects. The objects and their interrelationships are defined graphically, as are the Event Diagrams. The domain expert then defines the mathematical methods for each object. Auto-code generation into C/C++ is handled by Power Model. The applications for anomaly analysis, limit detection and trending are created with this tool. Using a single tool is desirable from a maintenance standpoint. When changes need to be made, they are made in the object environment, not at the C/C++ level.

The CBR KB consists of a set of stored cases in a library, where each case represents a unique prior failure. The CBR library constitutes a memory of past failures. When resolving a new failure, a solution is obtained by retrieving a stored case which is nearest (match and adapt) to the failure case. This approach is useful for problems that tend to recur, like Star Tracker attitude correction and attitude determination failures, and where failure reports are closed to a prior similar failure, when one exists. CBR assists anomaly resolution when there is a lack of reliable domain theory. Ideally we know how the Star Tracker should operate, but its on-orbit behavior indicates a lack-of-depth in our model. The CBR

KB represents experiential memory, which has been properly indexed to facilitate retrieval. Cases and models complement each other, as do rules and cases. Cases define performance behavior based on experience, but usually do not cover the entire domain problem space. Models cover the problem space between the cases. Hardware models are not yet in our system, but they are planned for final development.

CBR is more natural than rules for resolving particular types of control system problems. Cases of failures can be constructed relatively quickly and the validation effort appears to be faster than for a rule-based system, based on the experience in developing this system. The cases are entered through a user interface, which is menu driven. The case-based reasoning tool is similar to training a Neural Net. The case library consists of a number of stored cases, another set of unstored cases is used to optimize and validate the application. The first application developed was intended to identify the reason for a Star Tracker attitude correction failure. For HST, there are several hundred star tracker anomaly reports, each of which were reviewed for inclusion into the case library.

The index values are obtained from telemetry, the planning system, or are derived from telemetry. The tool builds a Cluster Tree based on the case library. The first release of this tool only accommodates a user interface, we had anticipated that the second release would have the ability to be imbedded in an automated system. However, due to a loss of funding, there may be no second release. The following work-around will mitigate this problem. The Cases will be ported into Microsoft Access, which will constitute the Case Library. The CBR tool will read the cases from Access and produce the Cluster Tree, which can then be translated into production rules, or Bayesian Nets and inserted into the OLES. Once they determine which Case matches the problem, the Case can be retrieved from Access and supplied to the Engineer, or to another OLES element. Implementing this approach, with the benefit of Access, will allow monographs and explanations to be supplied with each Case.

Validation

Verification and validation of the KB is essential. Verification encompasses things like circular rule chains and dead end rule chains. This effort has not relied on any specific software tool to perform verification, primarily because an adequate tool could not be found. Validation ensures that the KB can be used to correctly and accurately monitor, detect and resolve anomalies. Because of the wealth of HST historical data, validation is easier and resources need not be spent on constructing simulations to generate data. Validation is performed incrementally. For each new functionality added, appropriate telemetry to test out that function is extracted and input to the real-time inference engine which then initiates the process of anomaly detection and resolution. All validation telemetry data sets are saved in a library and are periodically used for regression testing to ensure that all elements of the system are intact.

If automation will be incrementally improved over the life span of the spacecraft, what metrics can be applied to this automated system to measure change over time? A previous study, (Dodhiawala, et al., 1989) indicates that the following metrics are necessary: speed, responsiveness, timeliness, and graceful adaptation. "Speed is the rate of execution of tasks. Responsiveness is the ability to stay alert to incoming events. Timeliness is the ability to react to and meet deadlines. Graceful adaptation is the ability to reset task priorities according to changes in work load or resource availability." Once a system is delivered, baseline measurements should be made, so that improvements with future releases can be quantitatively measured.

Automating Attitude Determination

Presently attitude determination is a labor-intensive process. The software is considered legacy software, it is accessed through a GUI resident on a UNIX operating system. The process is initiated manually and the user is required to make decisions along the process to achieve the desired result. The output products include a report, the attitude error and the command quaternion. If the error is sufficient to impact the up-coming science plan, then the new command quaternion is up-linked to HST and the spacecraft is maneuvered to null the error. The operator files a Problem Report, which involves paper products. This process requires continuous staffing, since an error in attitude can occur at any time. If the attitude correction fails, or if the Star Trackers did not behave as expected, then another Problem Report is filed and the operator searches for another opportunity to repeat the process. Our approach relies on network messaging to initiate the process as a function of the as-executed command timeline. Meta-level rules control processing decisions, the electronic archiving of the attitude and any problem reports, updating the star catalog for stars detected, and passing the command quaternion and attitude error to the next automation element.

The Search Scan Mode implementation implies that the stars detected in the Star Trackers may not constitute a complete set of stars within the field-of-view. Depending on pre-defined star grouping and patterns will thus result in a high failure rate for attitude determination. Instead the expected stars are extracted from the star catalog, ranked from brightest to dimmest, and matched to what was detected in telemetry. Each of the detected stars is corrected for sensor-induced distortion, which is a function of temperature, magnetic field influence and star intensity. The computed "true" H and V positions are then rotated into the Inertial frame for star identification. A triad matching of the stars is then conducted, to determine each Star Tracker boresight eigenvector. The stars are then corrected for proper motion. The boresight eigenvectors are combined and rotated into the inertial frame, which then yields the spacecraft pointing in the inertial frame. A pointing error results in the computation of a new Inertial Command quaternion. The Task Manager reviews the science plan and the spacecraft state and communication schedule, to decide when to up-link the new command quaternion. If the attitude determination fails, the telemetry is processed and translated into a Case. This new Case is then compared with the Case Library of attitude determination failures to identify the cause. When a match is found, the information is appended to the Problem Report. The Task Manager then decides when to conduct another attitude determination.

Autonomous Attitude Determination

Once the automated process is considered highly reliable, it will be possible to reformulate it into an automated procedure and install it on-board. The new flight computer, being installed in the 3rd Servicing Mission, may have sufficient memory to accommodate this task. The first step will be to size the memory requirements, then to convert the algorithms into C. This plan will provide a complete on-board response to attitude errors.

Summary

The HST reengineering effort is a multi-year project, whose goal is to substantially reduce the cost of operating and maintaining HST. The paper describes a methodology for assimilating the expertise of the Mission Engineers, using the domain of the attitude control system as the example [Attitude determination and the resolution of anomalies encountered during the process have been discussed. A multi-paradigm environment, consisting of rules and Case Based Reasoning has been found to be amenable to this task.], in a direct manner thereby avoiding the overhead of a large software and knowledge development staff and consequently allowing a reduction in the maintenance costs of the system.

The second key feature proposed and described in the paper is a low-risk architecture for autonomous operations. It is designated as Quasi-Autonomous. In this architecture, human interaction with the automated system is essential but is gradually replaced by autonomous operation as the system is "trained" by the human mentors. In this concept, one does not build an autonomous system *ab initio* instead it is evolved with time. The operators handle all new anomalies and are responsible for updating the rules and knowledge base once resolution is achieved. In the early phases, the system "recommends" appropriate responses, later it acts directly. The special tools and methodology described above are mandatory to absolutely minimize the "development" cost. Confidence in the system is maintained, procedures are validated without resorting to complicated simulations, and flexibility exists to accommodate changing/aging spacecraft conditions.

One aspect of this architecture is that it lends itself to the gradual automation of the spacecraft sustaining engineering functions and thus reduces engineering staffing requirements. The architecture therefore provides the prospect of reducing the staffing levels associated with sustaining engineering and system maintenance which have far higher staffing components than actual operations. While the staffing levels for operators does decline, the skill/expertise required increases enhancing the career potential.

References

Fish, D.G., 1999 "Balancing Manual Operations, Automation, and Autonomy in the Re-engineering of the Hubble Space Telescope Control Center". Proceedings, 2nd International Symposium on Spacecraft Ground Control and Data Systems SCD II.

Bradley, A. and Ryan, J., 1991, "HST Pointing Control System: Designed For Performance and Mission Operations". ESA Symposium on Spacecraft Flight Dynamics, ESA SP-326, page 231-238.

Bichindaritz, I., Kansu, E. and Sullivan, K. M., 1998, "Integrating Case-Based Reasoning, Rule-Based Reasoning and Intelligent Retrieval for Medical Problem-Solving". AAAI Workshop on Case-Based Reasoning Integrations, AAAI Press, Tech. Report WS-98-15, page 22-27.

Chen, H.and Wilkinson, L. J., 1998, "Case Match Reduction through the Integration of Rule-based and Case-based Reasoning Procedures". AAAI Workshop on Case-Based Reasoning Integrations, AAAI Press, Tech. Report WS-98-15, page 33-38.

Oertel, W. and Peterson, U., 1998, "Managing Episodes, Cases, Concepts, and Rules – an Integration Approach for Medical Problem Solving". AAAI Workshop on Case-Based Reasoning Integrations, AAAI Press, Tech. Report WS-98-15, page 120-125.

Siemens, R. W., Golden, M. and Ferguson, J. C., 1986, "Star Plan II: Evolution of an Expert System". AAAI-86 Proceedings, page 844-850, 1986.

Dodhiawala, R., Sridharan, N. S., Raulefs, P. and Pickering, 1989, "C. Real-time AI Systems: A Definition and An Architecture". IJCAI-89 Proceedings, Vol. 1 page 256-261.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 397

ISSN 0100-7386
Printed in Brazil

Space Technology

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp.398-402

ISSN 0100-7386
Printed in Brazil

The Magnetic Storm Predictor – MSP

Otavio Santos Cupertino Durão
Brazilian Institute for Space Research – INPE
12227-010; São José dos Campos, SP Brazil
durao@dem.inpe.br

Kim Leschly
Jet Propulsion Laboratory – JPL
4800 Oak Grove Dr.; Pasadena, CA 91109-8099 USA
kim.o.leschly@jpl.nasa.gov

Abstract

Magnetic storms[1] are a major concern for space activities. Electronic satellite failures (SEU), power grid impairments, increase in the atmosphere density and hazard for EVA are just a few of these concerns. This is particularly true for these days of the MIR, Shuttle and moreover the Space Station assembly and operation. These storms are hardly predictable from Earth or at the Lagrangian point, where WIND and ACE satellites are located. Therefore it is necessary to place a satellite at a more favorable orbit to monitor the phenomenon. The objective is to be able to predict these storms around three days in advance. This paper describes the Magnetic Storm Predictor – MSP, a mission designed to monitor magnetic storms and identifying those that will have an impact on Earth about three days prior it reach the Earth atmosphere.

Keywords: *magnetic storms, mission design, monitoring, forecast.*

Introduction

Magnetic storms are generated by two solar phenomena. The coronal mass ejection (CME) and the corotating interaction region (CIR). The first one is the massive eruption of tens of millions of tons of solar atmosphere travelling at the interplanetary space with speeds up to 2,000 km/sec. The latter is the sweep of high speed streams that corotate with the Sun and reach the Earth atmosphere. CME´s are more frequent during the solar maximum (about two per day) but still existent during the solar minimum as well (about one at five days intervals). CIR´s are more frequent during the descendent of the solar cycle. Obviously not every CME will cause a magnetic storm on the Earth atmosphere. Besides its intensity, two other factors will determine if it will have a significant impact on Earth (in fact the effects are not only at the atmosphere): i) its direction and ii) the structure of its magnetic field. Figure 1 shows images of CME´s occurring at the solar maximum of 1980 (August 18; source: High Altitude Observatory).

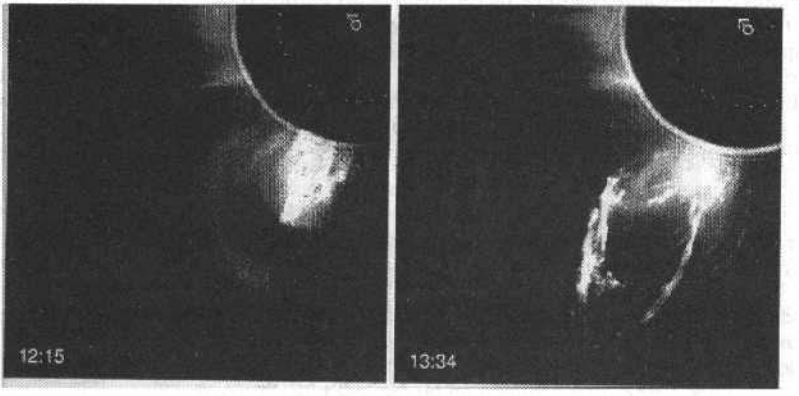


Fig. 1 Coronal Mass Ejection (CME)

However the observation of the CME´s from the ground or from LEO or at the Lagrangian point (where WIND and ACE satellites are located) is not sufficient for a reliable forecast of the magnetic storms. The Sun, as a background noise prevents a reliable observation of the CME´s directed toward the Earth. Thus, the forecast, at best, can be made just when the storm reaches the satellites placed at L1. This is just one or two hours prior it reaches the Earth, thus too late for any safety measure. On the other hand, observations made from Earth provide good sight of CME´s that are at an angle larger than 40° with the Sun-Earth line. Therefore the idea to place MSP at the ecliptic plane, drifting from Earth with an heliocentric orbit with a constant velocity (degrees per year), approximately at 1 AU from the Sun. Also,

in order to have a forecast of the storm it is necessary to measure the speed of the CME. This calls for two satellites, named MSP-1 and MSP-2 (in fact two satellites will also provide a 3D view for the scientific analysis of the CME`s[2], another purpose of this mission).

Magnetic storms are a hazard for space hardware and astronauts and its prediction will provide operation plan alternatives to avoid the damages it causes. Among known space problems related to magnetic storms one can enumerate: drop in solar array output due to high radiation

- increased SEU hits
- loss of RF communication/data transmission
- drop in altitude due to increased atmospheric drag
- loss of satellite tracking
- spacecraft charging and arcing
- electronic anomalies and loss of circuitry
- loss of spacecraft attitude control
- radiation hazard to astronauts

Some effects of the big geomagnetic storms of March and October 89 were:

- GOES-7; com-circuit anomaly (3/12), and loss of imaging (3/13)
- Japanese com-sat CS-3B; anomaly (3/17) and permanent loss of half of dual redundant command circuitry
- Japanese geostationary satellite GMS-3; severe scintillations (3/23, 1200-1430 UTC) and data transmission lost for one hour (3/23)
- Seven geosynchronous com-sats; orbital attitude anomalies which required 177 thruster firings to correct (March)
- GOES-5 and 6; seven SEU observed (October)
- DMSP; microwave transmission lost (October)
- TDRSS-A, C and D; SEU RAM hits – 56 total (October)
- INSAT-1B; loss of attitude control (October)

On the ground, known effects of magnetic storms are induction on high tension lines and power transformers. The nuclear plant transformer of a 1,200 MVA bank that exists on the Delaware river in New Jersey was damaged during the March 1989 storm. The loss of production from the plant cost up to US$ 400,000 per day during the 6 week outage[1].

Mission Design

Two satellites (MSP-1 and 2) will be placed in an heliocentric orbitat the ecliptic plane with a distance a little larger than 1 AU from the Sun. They will be drifting away from Earth, lagging it with a velocity of approximately 20°/year for MSP-1 and 40°/year for MSP-2. Therefore, after one year MSP-1 is at an angle of 20° with the Sun-Earth line, trailing the Earth, while MSP-2 is at 40° with that line and trailing the Earth as well. The operational life for the mission is two years. The satellites will monitor CME´s with a coronograph in a remote sensing activity, while in situ payload sensors in the satellite will detect the CIR´s when they are swept by it before reaching the Earth. Figure 2 depicts this scenario.

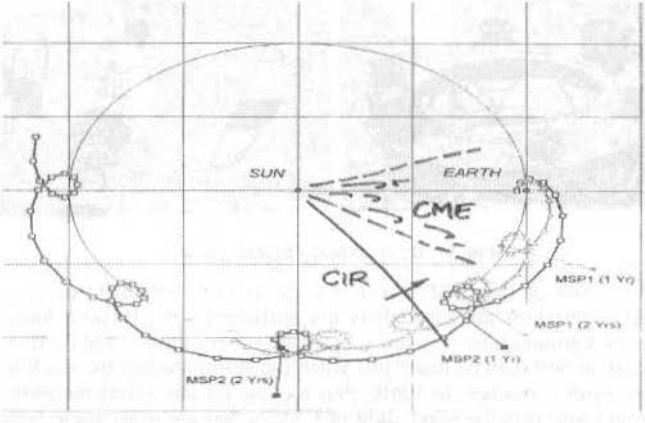


Fig. 2 MSP 1 and 2 and its trajectories

MSP is a mission designed to operate integrated with two other ones. A low Earth orbit imager and another satellite placed inside L1 equipped with in-situ fields and particle instruments, in order to make measurements of the magnetic field generated by the storms. These complementary missions are the Solar Mass Ejection Imager (SMEI) and Geostorm Warning Mission (Geostorm). SMEI consists of a white light all-sky imager which can image portions of CME´s to within 18° from the Sun-Earth line, with a resolution of ½ degree. It will provide one image per orbit at every 90 minutes. Its purpose is to do a final analysis on which CME´s forecasted by MSP as hazardous to Earth are going to reach the planet. Figure 3 shows MSP, SMEI and Geostorm.

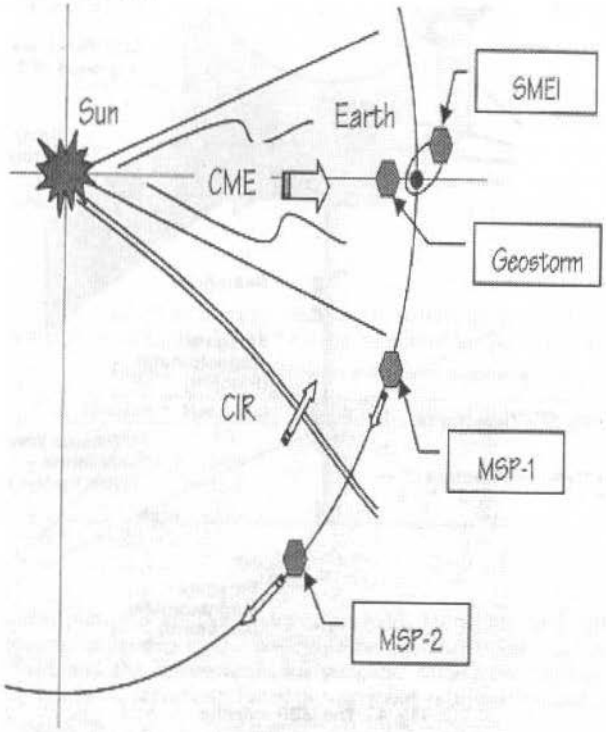


Fig. 3 MSP, SMEI and Geostorm

SMEI and Geostorm missions are already being studied by other institutions. MSP data will have to be analyzed integrated with SMEI and Geostorm. The purpose of Geostorm is to analyze the structure of the magnetic field of the storm in order to check if it is such that will interact with the Earth magnetosphere (big storms still may not affect the Earth depending on their structure). This final alarm will be given just a few hours prior the storm reaches the Earth, when Geostorm at L1 feels it (still better than what is provided by ACE or WIND). Therefore, the operational integrated plan for the three missions combined is that MSP is able to warn for the storm about 3 days in advance; SMEI will indicate if the storms is going to reach the Earth and Geostorm will give the final alarm a few hours prior it arrives. ACE or WIND may be used instead of Geostorm, however loosing a few hours warning. MSP by itself will be able to do research regarding the CME´s and provide an alarm for the storms about 3 days in advance, however without the same precision that would be obtained with SMEI and Geostorm (or ACE or WIND).

The MSP satellite

The MSP satellite (Figure 4) is composed by three main parts: the satellite bus (platform), the paylods and the kick stage. The kick stage and its propellant (about half the total mass of the satellite) will allow for autonomous insertion into its final orbit, independently of the launcher. Therefore, that is no need that injection is made by the last stage of the launcher. MSP, then, can be launched as a piggy back payload, with considerable saving in costs. The initial orbit is GTO and there is no launch window constraints.

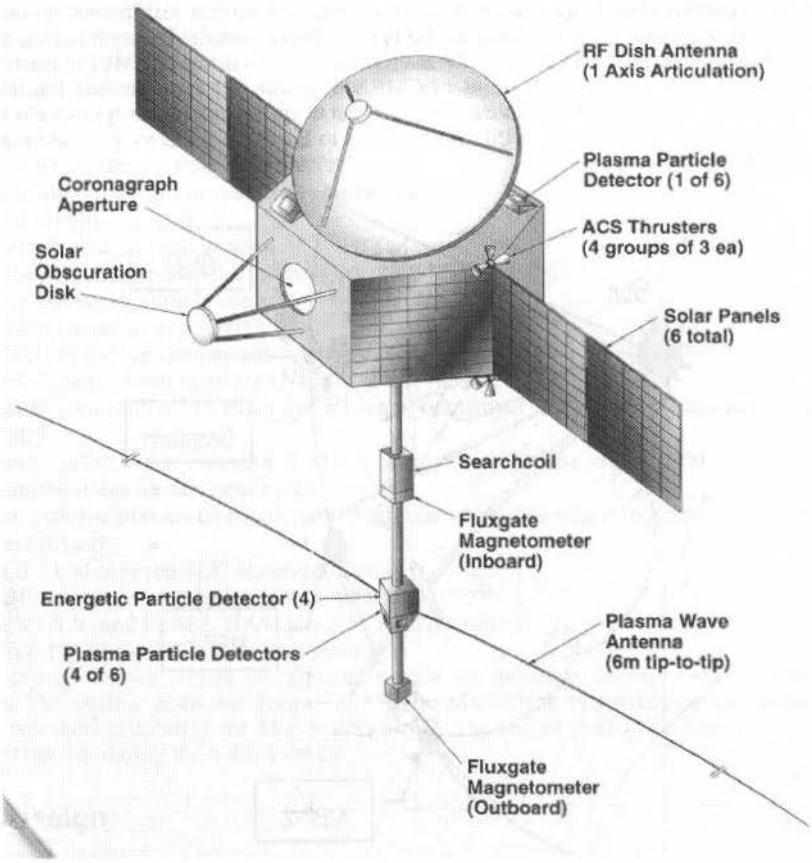


Fig. 4 The MSP satellite

The satellite will use miniature fields and particle instruments currently being developed at JPL for Solar Probe, and a miniature coronograph. The bus will use microspacecraft design with 3 axis stabilization. Total satellite mass is aimed at 100 kg. It will operate under autonomous monitoring of CME´s and CIR´s (beacon mode) to indicate CME/CIR activity level once a day. It will perform full data dump in case of potential major geomagnetic storm.

The MSP bus (platform)

The platform is based on a microspacecraft design, low cost, multi-functional, with 3 axis stabilization. It is an hexagonal prism with 50 cm. tip to tip and 25 cm. high, low cost iso-grid structure with deployable instrument boom and solar panels, and a one axis articulated dish antenna. Data rate is at 1 Mbyte/sec. (internal) and 100 kbytes per snapshot; data storage is 1 Gbits (aprox. 100 snapshots). The processor is RH-32 or RAD6000 or equivalent. RF communication is in S-band and dish antenna. Pointing is aimed at 10 arcsec./min (control) and 1 arcsec. (knowledge).

Payload

MSP payload will be the miniaturized Integrated Space Physics Instrument (ISPI) with low mass/power (16 kg./16 W goal) to make in-situ plasma and field measurements and coronograph imaging. The ISPI architecture is based on an interchangeable sensor suite, allowing it to be available for a multitude of space missions (a subset of ISPI will fly on the STRV-2 and DS1 missions). The full complement (9 sensor types) is baselined for Solar Probe. Table 1 shows the full set of sensors for ISPI, including its masses, power and data rates.

Table 1 ISPI Mass/Power/Data Rates

Sensor/Element	Mass (kg)	Power (W)	Data (kbps)
Fluxgate Magnetometers (2)	0.3	0.4	1.2
Plasma Wave Sensors	0.5	0.4	9.6
Proton Plasma (8)	0.8	1.5	20.0
Ion/Electron Plasma (2)	3.1	3.4	20.0
Neutral Particles (3)	0.3	0.4	0.8
Energetic Particles (2)	0.4	0.5	4.0
EUV Imager	2.0	0.8	30.0
Magnetograph	1.8	1.5	30.0
Coronograph	2.4	2.0	2.0
ISPI Data Processor	2.0	6.0	0.5
ISPI Power Conv./Distr.	0.5	2.0	0.0
Total	14.1	18.9	118.1

When compared with other set of instruments that flew in other missions, ISPI is dramatically advantageous, as is shown in Table 2.

Launcher

MSP 1 and 2 are to be launched as a piggy back, simultaneously by an Ariane V launcher. As 100 kg. satellites they fit perfectly well in the ASAP (Ariane Structure for Auxiliary Payload).

Table 2 ISPI comparison with other payloads

Mission	Mass (kg)	Power (W)	No. of sensors
Galileo	59	56	6
Cassini	140	130	6
Wind	100	70	4
ISPI	12	11	9

Costs

The MSP total costs (platform and kick stage, payload, launching and operation) is significantly lower than similar missions proposed in the past, due to miniaturization and autonomy. Key enabling technologies are the ISPI and the microspacecraft concept. Total cost for the mission, including the development of the two satellites, launching for both, two years of operation and a 30% budget reserve, is estimated in 40M US$.

Conclusions

The MSP mission is going to be proposed within the announcement/mission of opportunity for the NASA STEREO (Solar-Terrestrial Relations Observatory) Mission[3,4]. This mission aims primarily for the observation of the CME´s, and its scientific understanding, and also for alarm against magnetic storms. MSP has established a symmetric set of priorities and aims to be included as a precursor for the more ambitious STEREO Mission. In this way there will also be the opportunity to explore the next solar maximum, due to the advances already done in the development of ISPI. This may shorten the launching date for the MSP mission.

References

[1]Magnetic Storms; Tsurutani, B.T.; Gonzalez, W.D.; Kamide. Y.; *Surveys in Geophysics* 18: 363-383, 1997.

[2]Solar Tomography; Davila, J.M.; *The Astrophysical Journal*, 423: 871-877, 1994 March 10.

[3]The Sun and Heliosphere in Three Dimensions; *Report of the NASA Science Definition Team for the STEREO Mission;* edited and published at The John Hopkins University Applied Physics Laboratory, Laurel, Maryland, 1 December 1997 (also available at sd-www.jhuapl.edu/STEREO)

[4]STEREO Mission Design; report prepared by John Galloway/740; NASA/GSFC; June 26, 1998.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp.403-410

ISSN 0100-7386
Printed in Brazil

New Generation of On-Board Computers for the International Space Station

Günther Brandt
DaimlerChrysler Aerospace
Huenefeldstrasse 1 – 5, 28199 Bremen
guenther.brandt@ri.dasa.de

Abstract

A new computer family for the International Space Station (ISS) has been developed in Europe which copes with the various demands of the distributed data management architecture of the Station. Two basically different types are available:

1. *Fault Tolerant Computer (FTC) for safety critical tasks like station and Space vehicle control and navigation*
2. *Payload processor for experiment monitoring and control in all payload locations*

The DaimlerChrysler Aerospace (Dasa) Fault Tolerant Computer is designed for space application where high reliability, availability and radiation-tolerance is required. It is a modular computer that provides high processing power to the user and can tolerate up to two non-simultaneous faults. For communication purposes the FTC is equipped with six MIL-STD 1553B Interfaces and is prepared to support the Synchronous Packet Transfer (SPT) protocol as used for communication onboard the International Space Station (ISS).

The FTC consist of identical computers (hardware and software), each in a separate housing, called Fault Containment Region (FCR), operating simultaneously and executing identical User Application Software. Dependent on the required fault tolerance, up to four FCRs (for two fault tolerance) can be combined to one FTC. The design is based on the so-called Byzantine Failure Algorithm.

In the vital domain of payload control and data processing, the Standard Payload Computer (SPLC) represents a novel solution in space technology. It is confronted by new and challenging requirements, particularly concerning extended lifetime, radiation, reconfigurability, maintainability and logistics aspects as well as the need to reduce development duration, risk and cost.

The SPLC is a modular computer, configurable according to the payload developer's needs from a shopping list of space-qualified standardised components.

The paper will describe concepts and features of both computer types, both based on same design principles and the same processor chip.

Introduction

Life cycle of commercial electronics and in particular microprocessor chips and subsequent hardware designs became shorter and shorter in the past 10 to 20 years. In fact meanwhile the computer performance doubles in about half years time span. The gap between commercially available technology for terrestrial use and space born technology became wider and wider in the recent past.

Commercial "off the shelf" products were used in some specific cases, however the unique space environmental constraints do not permit to integrate such products in systems with safety critical functions and high reliability requirements. Main reason for this fact is availability of high reliable, radiation tolerant EEE parts, in particular processor and memory chips. Therefore for each particular space mission unique design solutions are being developed which are the main driver for schedule and cost.

As a consequence of unique onboard data management design solutions the software programs are special, single use products and associated ground support equipment and software design development environments need to be developed.

A way out of this trend needs to be found. It must be a way that ground systems and onboard software can use commercially available products and onboard hardware architecture is structured to be modular and configurable to cope with multiple applications and to be capable of interfacing with commercial hardware and software. As an example an astronaut or cosmonaut will not accept to work with MS-DOS as a user interface, if they are used to work with windows 98. For safety critical application a fault tolerance rather than fault avoidance is required. This challenge to cope with the said above is described in the following.

New Design Concept

A technology gap between space avionics and terrestrial electronics in particular for computers will always exist since space avionics has to withstand the effects of radiation and therefore certain radiation

sensitive technologies can not be used in space. However, a design concept which deviates from previous practice of special development can minimise the performance gap and what is more important can minimise program cost and design and development schedule for a particular space mission. This new design concept which has been applied consequently for the first time for the European Space Station computers provides for the following key features:

- Modularity on two levels; first, board in EURO size for basic functions and second on a hook-up board on top of the basic board, the so-called Mezzanine board
- Open computer architecture
- Functions accommodated on one board are selected such that various board combinations are possible
- Rigorous implementation of interface standards:
  - VME (industrial internal computer bus standard)
  - MIL 1553B
  - Ethernet
  - RS 422
- Usage of a commercially wide spread processor (SPARC) in a radiation hardened version. However, CPU board architecture is designed such that processors available in the future can be accommodated with minimal design changes on computer level
- Usage of standard commercial Real Time Operating Systems to ensure compatibility with commercial software products, ground equipment and available software development environments.
- Separation of input/output functions from **C**entral **P**rocessor **U**nit (CPU) load, that means peripheral interfaces with built in intelligence to minimise CPU load and to maximize data throughput as needed by application.
- Software separated into layers with defined interfaces in between. Typical software layers are:
  - Board support package (HW to SW interface)
  - Operating System with hardware drivers
  - Interface protocol software for external communications

These new design concept principles have lead to two computer types:

1. A Fault Tolerant Computer for safety critical applications
2. A Modular Payload Processor for ISS experiment facilities

Fault Tolerant Computer design for safety critical applications

The **F**ault **T**olerant **C**omputer (FTC) is designed for space applications where high reliability, availability and radiation-tolerance is required, It is a modular computer that provides high processing power to the user and can deal with up to two non-simultaneous faults. For communication purposes the FTC is equipped with six MIL-STD 1553B Interfaces and is prepared to support the **S**ynchronous **P**acket **T**ransfer (SPT) protocol as used for communication onboard the International Space Station.

The FTC consists of identical computers (hardware and software), each in a separate housing as shown in Fig. 1, called **F**ault **C**ontainment **R**egion (FCR) operating simultaneously and executing identical User Application Software. Dependent on the required fault tolerance , up to four FCRs (for two fault tolerance) can be combined to on FTC.

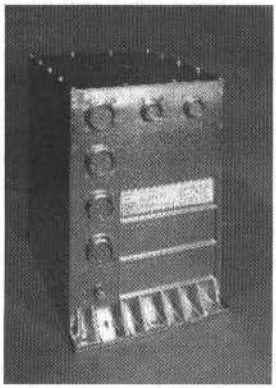


Fig. 1 Fault Containment Region

The design is based on the so-called *Byzantine Failure Algorithm* [1]. According to this theory, a computer can isolate F arbitrary (*Byzantine*) faults, if the following conditions are met:

- The computer must consist of (3 F + 1) Fault Containment Regions (FCRs)
- The FCRs must be interconnected through (2 F + 1) disjoint paths
- The data must be exchanged (F + 1) times between the FCRs and voted according to the Byzantine algorithms
- The FCRs must be synchronised to provide a bounded skew

This concept has been rigorously and consistently implemented in the hardware and software design of the FTC. For F = 1 (four FCRs configuration), i.e. it can deal with one such completely arbitrary fault at a time. On detection of such a fault the error is masked and the FTC continues operation, if the fault is of temporary nature, e.g. caused by a Single Event Upset. If the same fault occurs n-times within a timeframe of 100 ms, the faulty FCR is automatically isolated (deactivated) and the FTC continues operation in a 3-FCRs configuration. In this configuration it is still fault-tolerant with respect to a deterministic fault by performing majority voting, thus meeting the *two-fault-tolerance* requirement in this sense.

The FTC System

The FTC Design consists of up to four identical FCRs. Each FCR is accommodated in a separate housing. A separate power inlet and power switching interface support the configurability of the FTC with one, two, three and four FCRs. Each FCR box is exchangeable and can be reintegrated into the FTC during operation.

All FCRs of one FTC are connected to each other via a cross-strapping harness for:

- Exchange of data to be voted
- Synchronisation of software processing and data distribution by a fault-tolerance clock
- Fault-tolerant majority isolation (reset) of a faulty FCR by the remaining FCRs (4-FCRs configuration)

These cross-strapping connections are realised by galvanically isolated point-to-point connections to avoid error propagation between the FCRs.

Figure 2 shows one complete FTC configuration with four FCRs, the cross-strapping connections and the principle connections to the MIL-STD 1553B bus system.

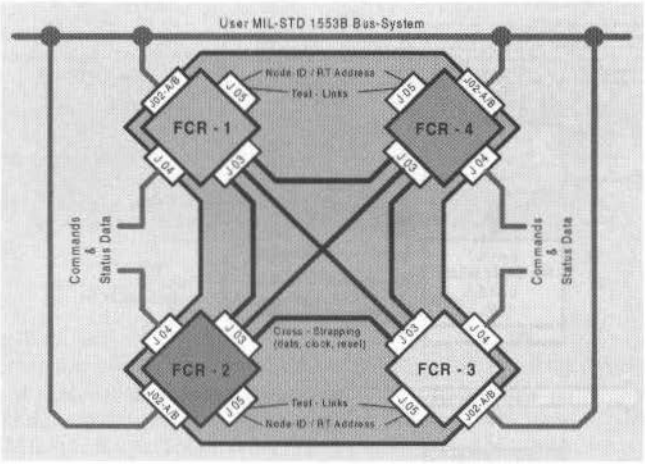


Fig. 2 FTC Cross-Strapping Connections

Each FCR can be connected to up to six different MIL-STD 1553B buses providing up to two **R**emote **T**erminal (**RT**) functions and four **B**us **C**ontroller (**BC**) functions, where one BC function will be active per FCR during nominal operation. The FTC provides for different data acquisition, commands and voting strategies, configurable to the user's needs. In case of an FCR fault, the BC function of this FCR will be taken over by one of the remaining FCRs, according to a pre- defined order. Each FCR provides an input for its own RT address (adjustable via connector wire-bridges); both RTs of one FCR have the same address, this RT address must be different form those of the other FCRs.

The FCR Architecture is realised by a separation of the processor executing the **U**sers **A**pplication **S**oftware (**UAS**) from those processors responsible for the fault management and the management of the MIL-STD 1553B bus interfaces in order to release the user from the fault management functions.

This results in a three layer architecture of the FCR as outlined in the Figure 4 which gives an overview about the separation between Fault Management Layer (incl. MIL-Bus Interface) and Application Layer.

Figure 3 depicts as an example application the detailed connections of the FTC to the MIL-STD 1553B bus system.

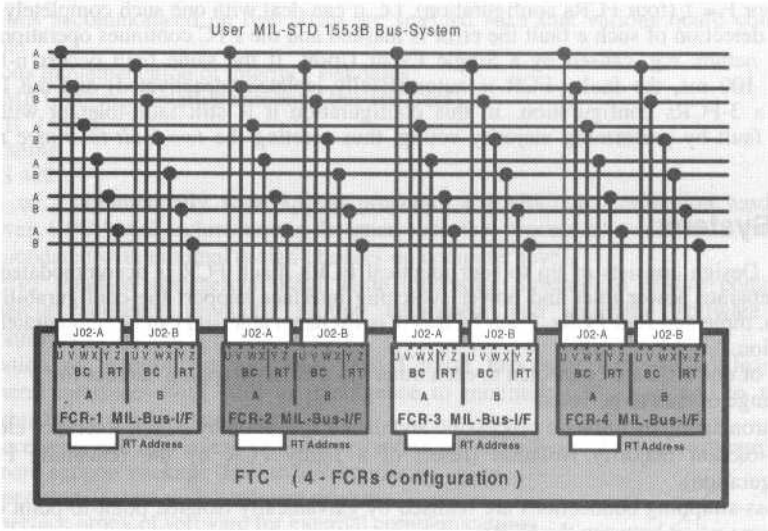


Fig. 3 FTC Connection to the MIL-STD 1553B Bus System

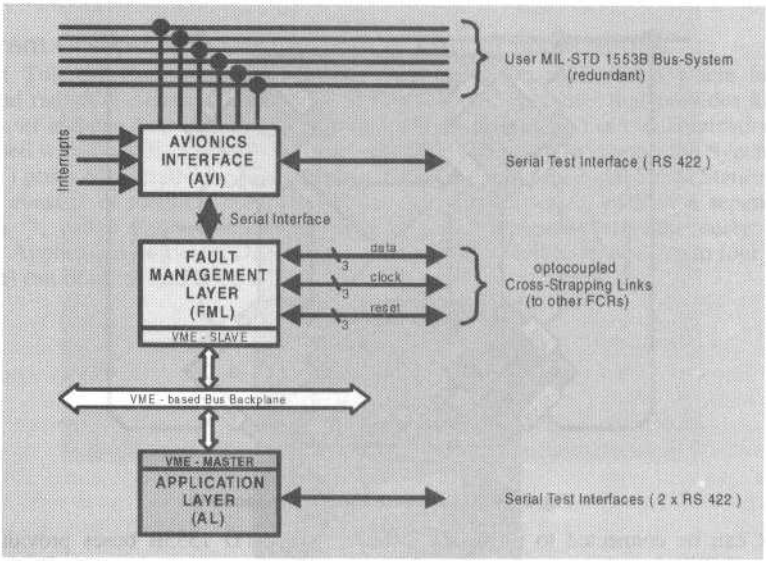


Fig. 4 Three Layer Architecture of the FCR

- The design of the **A**pplication **L**ayer **B**oard (**ALB**) is based on the radiation-tolerant SPARC-compatible chipset **ERC32** and uses the standard off-the shelf operating system **VxWorks** which provides the platform for the Users Application Software.

- The **F**ault **M**anagement **L**ayer (**FML**) PCB and **AV**ionics **I**nterface (**AVI**) PCB are designed on the basis of the Transputer **T805/20 MHz**. The FML PCB provides the fault management functions and the galvanically de-coupled interfaces for the FCR cross-strapping. The AVI PCB provides the physical interface to the user MIL-STD 1553B bus system.

For test purposes the ALB and the AVI-PCB provide bi-directional serial test links for software loading, debugging and failure injection.

Standard Payload Computer for universal applications

The **S**tandard **P**ayload **C**omputer (**SPLC**) represents a novel solution in space technology to fully meet the requirements outlined in the introduction and concept description. The SPLC resides between the payload and the ISS data management system as shown in Fig. 5. It puts an end to the costly and time-consuming traditional approach of the individual, proprietary development of payload computers for each new payload.

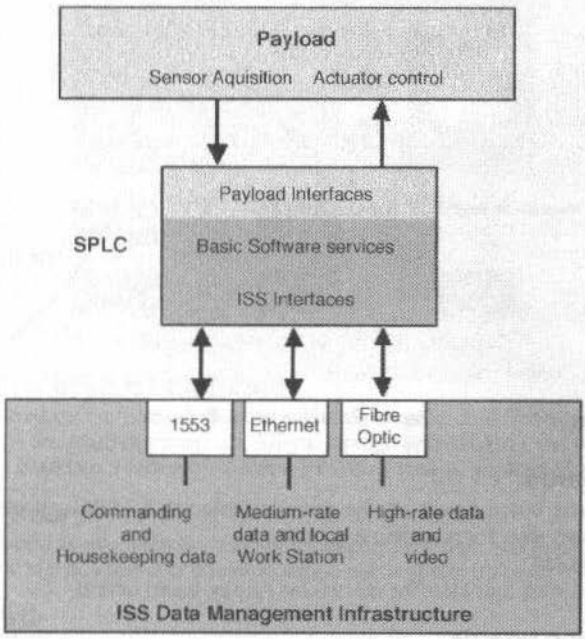


Fig. 5 SPLC with ISS data interfaces

The SPLC key features are:

- A modular computer concept configurable according to the payload developer's needs from a shopping list of space-qualified standardised computer components.
- A verified interface software which takes care of the communication between payload and ISS based on the MIL 1553B payload bus and the CCSDS packet protocol.
- A standardised and uniformed software development environment and ground support equipment.
- The standardisation is based on the open VME bus-standard and the commercial operating system VxWorks, which allows for full growth capability. In particular independently developed non standard components which are not provided by the SPLC shopping list can be integrated and also development of application software can be performed on fully compatible commercial products.

This concept allows for a totally new approach in payload computer development, significantly reducing development duration, cost and risk as well as providing the basis for an efficient, low-cost spares concept and maintenance approach for ISS payload computers.

SPLC Concept

The SPLC is based on the wide-spread industrial standard VME bus. It comprises VME boards and mezzanine (piggy-back) boards. The principal is depicted in Figure 6. These boards have been specifically selected and designed such that from these the widest possible range of payload computers can be configured.

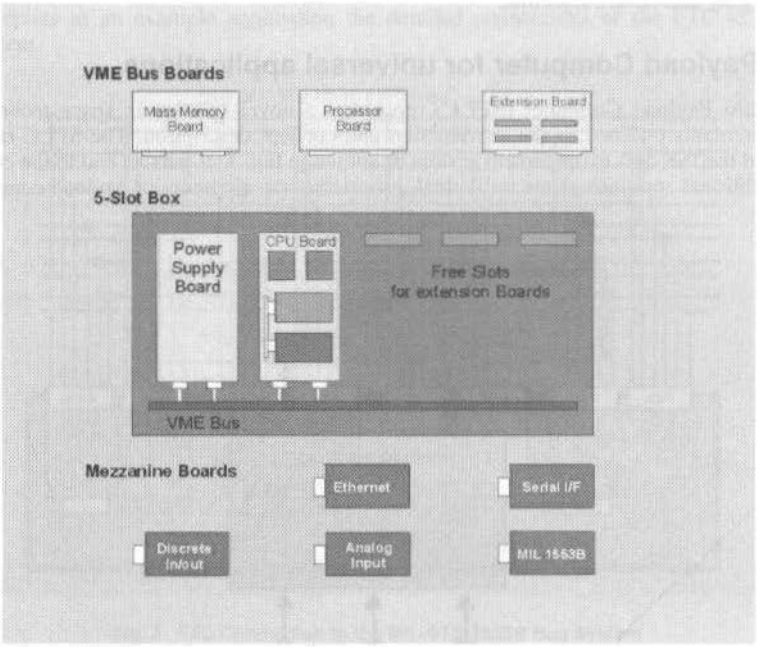


Fig. 6 Principle SPLC Concept

Available VME boards

- Central processor board based on the radiation-tolerant SPARC chipset ERC32 with 6 serial interfaces and two slots for mezzanine boards
- Mass memory board
- Extension board with four slots for mezzanine (piggy-back) boards

Available Mezzanine boards

- MIL 1553B interface board
- Ethernet interface board (IEEE 802.3)
- Serial I/F board (RS 422 and RS 485)
- Discrete 12 channel input/output board
- Analogue input board (8 channels)

These boards are fully interchangeable on all mezzanine slots, both on the extension and on the main processor board.

Equipped with the mezzanine MIL 1553B interface board (as data interface to the Space Station) and any of the other mezzanine boards according to the payload's internal interface requirements, the central processor board represents a single board computer which alone already meets the needs of many payloads with limited complexity.

Payload computer configurations for nearly any other payloads with more complex functional requirements can be build by adding VME extension boards with appropriate mezzanine interface boards and/or the mass memory board from the SPLC shopping list.

Housing

The SPLC standard housing is a lightweight 5-slot or 8-slot aluminium box with bolted upper cover plate as depicted in Fig. 7. It contains the VME-backplane and power supply. The same housing is being

used for the FTC. It is designed for conduction cooled VME boards. Of course, other housings customised to a specific application are also possible.

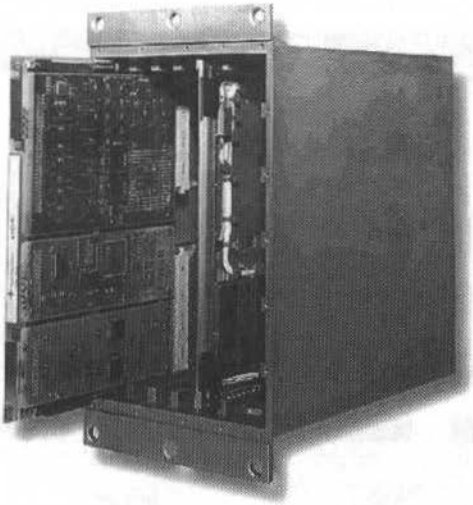


Fig. 7 5 slot standard SPLC housing

Environmental Design

The SPLC is designed for ISS pressurised and vacuum environment and withstands Space Shuttle and Ariane launch vehicle transportation environments. Depending on the individual application in vacuum, a thermal radiation system may be needed to be attached to the box surface.

VME Bus Central Processing Unit Board

The CPU board is based on the ERC32 chip set which is a space qualified version of the SPARC V7 processor. It also contains the main memory, a 8 Mbyte storage area (RAM), the 4 Mbyte EEPROM, the VME bus controller chip based on a radiation tolerant FPGA as well as six RS422 serial interfaces.

The Mass Memory Board

The mass memory board is available for storage of experiment data and application software. It can be accessed through the VME bus and is immune to single event upsets up to 36 MeV cm^2/mg.

The Extension Board

The Extension Board is a VME bus slave board supporting the A32, D8 memory model as well as interrupt handling. The board contains four I/O mezzanine board slots. Up to 32 Extension Modules may be integrated into one VME bus system.

Mezzanine Boards

Five different mezzanine boards are available. These are a MIL 1553B board, an Ethernet board, a serial I/F board, a discrete 12 channel input/output board and an 8 channel analogue input board. The boards are fully interchangeable on all mezzanine slots and feature local microcontrollers. The mezzanine bus is an open standard, allowing payload developers to build their own mezzanine boards.

SPLC Software

The SPLC is provided together with the Operating System Software, the Board Support Package and a Basic Software Package. The real time kernel of the OS is VxWorks V5.3. The Basic Software comprises all necessary interface software required for communication with COF and the US Lab (1553B, CCSDS) plus other support functions such as file transfer, monitoring, etc.

Software Development Environment and EGSE

For software development, checkout and test of the SPLC a combined SDE/EGSE is provided. It also comprises a script-based test tool for acceptance tests as well as interface testing and simulations, such as

ISS interface handling. It is built-up in a VME crate which provides spare slots for integration of stimuli/measurement boards to support functional and performance tests of application specific SPLCs.

The available range of components described above is shown in total in Fig. 8.

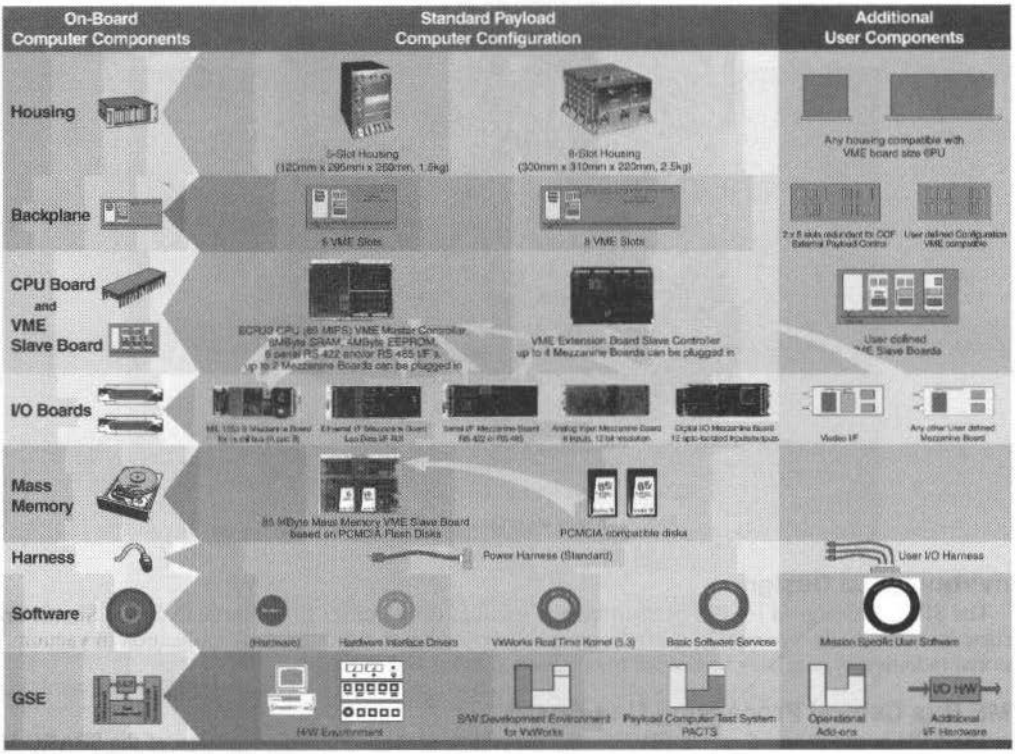


Fig. 8 Available components for the modular SPLC

Summary

Space electronics will also in future not follow with the same pace the technology evolution of terrestrial electronics. The physical environmental conditions in space will even prohibit certain technologies applied for EEE parts manufacturing. However architectural design of a computer can be done such that new technologies can be integrated and made use in space more easier with less development effort. This approach has been persued in the designs of the described Space Station computers. More advanced processor chips (e.g. power PC), memory chips with higher capacity or new input/output devices can be accommodated easily in the future as those parts become available for space applications.

References

[1]Lamport, Shostak and Pease: "The Byzantine General's Problem"

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp.411-415

ISSN 0100-7386
Printed in Brazil

Achieving the Ultimate in Cost Effective Ground Systems for Schools

Curtis Fatig
NASA/GSFC/Orbital Sciences Corporation
GSFC/440.8/T-12F, Greenbelt, MD/USA
curtis.fatig@gsfc.nasa.gov

Wynn Watson
NASA/GSFC
GSFC/441.0/Building 3, Greenbelt, MD/USA
wwatson@hst.nasa.gov

Craig Bickford
NASA/GSFC
GSFC/598.0/Building 25, Greenbelt, MD/USA
cbickfor@pop500.gsfc.nasa.gov

Abstract

The universities and public schools provide the people resources for our future spacecraft, ground system, and political leaders. Putting a cost effective control center in these classrooms will provide our future leaders with a greater understanding of satellite and space operations, increase their math and science skills, and give the general public a chance to be a part of the exciting space program. To achieve this goal a new shift in operations is needed to reduce the costs in control center operations. The remote control centers will allow for remote command and telemetry functions and eliminate the need for redundant equipment at each site. This paper will define one cost-effective control center approach that provides control center capabilities at student's desktop or in a small classroom in the educational facility.

Introduction

The schools will participate in a cooperative effort with experiments on the International Space Station (ISS) or use older satellites that are no longer needed by their respective organization. The schools will work together, with a single school being identified as the lead, to divide the responsibilities of spacecraft operations and to provide a transition path from older to younger students. Coordinators provided by the various space agencies and companies will assure proper and responsible spacecraft and experiment functions are adhered too. Due to the nature of this program the universities and schools will not be limited geographically. In fact, there are added social benefits if students from various nationalities participate in the same spacecraft or ISS experiment.

Approach

A truly cost effective control center provides control center capabilities at the student's desktop or in a classroom room in the educational facility. Current technologies can be used to communicate between existing networks at the remote and central facilities without costly modifications. The control center would provide, from a standard desktop computer at the student's location, all command, telemetry, and archiving functions. This would be accomplished in the following ways:

- -The telemetry would be transported, from the ground station, JSC, or other site, to the student using Information Sharing Protocol (ISP) over the Internet/Intranet. ISP is an existing protocol, in use by JSC, KSC, and HST, for transporting telemetry and displays over the Internet/Intranet. An individual student could log into the ISP/Web server using a custom web browser to look at predefined displays or create custom views of the data.
- -The telemetry would be archived at the ISP/Web server located at the lead school control center. This data would be saved automatically for retrieval and data analysis at a later time and is available to all students participating in the program.
- -Commands are sent through an encrypted IP tunnel over the Internet/Intranet to the uplink station (i.e., a Ground Station). The ground station would check the commands for completeness prior to uplinking to the experiment, whether it is on the space station or a free flyer. The commands will be automatically uplinked once a forward link is established with the experiment.

The HST functions that apply to the 'Cost Effective Ground Systems for Schools' are; web server for electronic documentation, web-based data retrieval and analysis tools and remote commanding

architecture. The knowledge gained through the HST program can be applied to the 'Cost Effective Ground Systems for Schools' to allow the smaller organizations the same level of data support at a reduced cost. The cost structure and flexibility for the remote control centers will allow for more students to directly participate in on-orbit scientific experiments.

Goals

A truly cost effective control center provides control center capabilities at the student's facility by:

1. Allowing for using low-end computer systems for the student's machine.
2. Use a combination of Commercial Off-The-Shelf (COTS) and custom software for accessing data and sending commands.
3. Providing data access and command capability via remotely shared servers.
4. Servers automatically providing the software needed by the student's machine.
5. Maintaining low overhead using existing facilities and donated equipment.
6. Provide lesson plans for elementary, middle school, and senior high school teachers.

Each of these items taken individually will not achieve the desired goal. All items must be implemented in total for the full cost savings to be seen.

Allowing For Using Low End Computer Systems For Student's Machine

The cost effective ground systems for schools will support a limited number of Operating Systems (OS) on the student machines. This will allow for fewer ports of any software required for the command, telemetry, and archiving systems. The student machines will be based on current home/office products. Current plans would be to support UNIX, Windows NT/98, and Mac OS/Rhapsody. This will cover 94% of the available OS's that reside on student machines.

Use a Combination of Cots and Custom Software for Accessing Data and Sending Commands

The use of COTS will be limited to a web browser. This saves long term costs associated with upgrading and licensing COTS for the life of the mission. The custom software will be built, tested and supported in a maintenance mode once the software is certified for use. Maintenance of the custom software will be limited to assuring compatibility with the upgrades to the OS's supported. The time between major OS releases is generally two to three years and this can be combined with other software support from other programs in order to reduce costs.

Providing Data Access and Command Capability Via Remotely Shared Servers

To protect the satellite or experiment from harm, command capability must be restricted to only authorized students. Command functions would reside on a separate server that would be administered by contractor staff and protected by firewalls; The students that have approval to send commands would be issued a Secure ID card. They would be required to enter the one-time password provided by the card as well as their username and password to log into the command server. The commands would be sent from their local workstation to the command server through an encrypted tunnel between the two systems. The command loads and individual commands will be viewable by the command student. The command subsystem will provide the student with a list of commands and their description. The student will select the appropriate commands or command loads from the list and send the commands to the command server and set a time window for the commands to be uplinked.

Software on the command server will perform integrity checking on the commands and verify that the commands will not violate any spacecraft constraints. The system will also allow the command student to reorder the sequence of commands, delete commands, and upload new commands that will be transmitted when a forward link is available within the time window that was set to uplink the commands. If there are any problems with the uplink of the commands the system will notify (i.e., page) the appropriate personnel to investigate the problem. Each student will have the capability to control his/her individual experiment on any space-based platform. Health and safety of the spacecraft or ISS would be the responsibility of one group.

For data access, the students will connect via Internet to a telemetry server located at the same school as the command server. The telemetry server will require the students to enter their username and password and access the system from a set of predetermined IP addresses on nonstandard IP port numbers. The session will be encrypted via Secure Socket Layer (SSL) compliant browser and server software. The telemetry server will provide a real-time stream of telemetry when available to the student's machine. Students can define telemetry pages to view and save on the system for future reference. The student will have the option of saving the page definitions in their private or shared directories so that

other students can use them. The telemetry system will have a short-term archive that maintains the most recent 30 days of data for retrieval at any time. This will allow the students to monitor the experiment at their convenience and not only when telemetry is available. Both the real-time and archived telemetry can be down loaded to the student's workstation for further analysis. The telemetry system will allow for a few key telemetry monitors to be analyzed with predetermined states. If the states do not match with the current real-time telemetry, the telemetry system will notify (i.e., page or e-mail) the appropriate student. The student can change the telemetry display pages at any time by uploading and activating the files to the telemetry server.

Commands and command loads that are on remote command workstations or on the centralized command server will be encrypted via Pretty Good Privacy (PGP) and sent between the systems via encrypted tunnels. A centralized agency key server will be used to maintain and update the PGP on a frequent basis. The telemetry will only be protected through the use of SSL during transmission between systems. The system will allow for students to encrypt data in their personnel directories with their private keys and to put data in group accessible directories accessible by students whose private key is among the list of private keys that are allowed. NASA is currently developing a key infrastructure that could support the key management.

As new students are added and other students are deleted to the system, it will be the primary teacher responsibility to determine what access is allowed, to provide username, password, and encryption keys. The telemetry and uplink systems will provide the primary teacher with a Web page to configure the access to the systems for changes in use profiles.

Servers Automatically Providing the Software Needed by the Student Machines

A student will log (username and password protected) into the systems server using a SSL compliant Web browser, the server will verify the machine's OS and determine what applications will be needed to support telemetry and archiving functions. The server will suggest, if needed, downloads for the student that will be needed for viewing and manipulating the data. A student could either download the applications, or proceed into the main student page which will have links to the various telemetry pages, archived products, mission timelines, and historical events. Once the student enters the student's main page, they will have access to all the data. Each student will have a profile maintained on an authentication server. This profile will determine what access each student has and log the student's activities. These logs will be used to troubleshoot any problems with the systems. Each student will have a unique write area where they can temporarily store files. The main archive area will not allow students to have write privileges.

Maintaining Low Overhead Using Existing Facilities and Donated Equipment.

To maintain a low overhead, the use of classrooms within the educational facilities and finding business partners to help defray the cost are a must. The classroom space often can be found in common areas of a school (i.e., libraries), a single room dedicated for the control center, or use of the schools computer center configured for dual use. Business partners can be related to computers or aerospace companies, however the best resources are those close to the school that can see their business or name recognition increase. Business can donate old equipment that can be resold or used in the control center, money for direct purchases, or discounts that can be used for purchases. Another way, used in many parts of the Untied States, is to provide part of every sale as a donation to the school. Business or other vocational schools can also provide volunteers to assist in the control center as system administrators, system maintenance, and classroom assistants.

A current program at the college level is the Bowie State University Satellite Operations and Control Center (BSOCC) and is a joint venture between Bowie State University and Goddard Space Flight Center. This unique model program, supported by Allied Signal Technical Services Corporation, creates an orbiting satellite operation and control center on a university campus staffed primarily by undergraduate students.

Provide Lesson Plans for Elementary, Middle School, and Senior High School Teachers

One item that is missing from many programs, is one that provides a structured learning approach throughout the student educational years. Once in college more detailed training is received and is will covered in most developed nations. This paper not only addresses this issue, but also will provide a basic program for a six-week lesson at each United States grade level (1^{st} though 12^{th}) that can be adapted for other nationalities.

An example at the college level is the BSOCC. The BSOCC has been operating the SAMPEX spacecraft, after it's operations were completed by NASA and has been so successful that its useful

mission lifetime has been extended. It has been included in the Solar Max Campaign to continue to provide its scientific data through the next period of maximum solar activity in the years 2001-2003. The BSOCC is expanding their program by training more students simultaneously, offering a curriculum concentration as part of a baccalaureate degree program, and being able to serve as a training and educational facility in space technology to students at other institutions.

Structure/Networks

The systems and networks supporting the cost effective control center provide the following equipment for command and telemetry.

1. One to many commercial desktop computers for students at all locations.
2. Hard disk space for long-term storage at the central school.
3. Hard disk space for short-term storage at the central school.
4. A dedicated commercial desktop computer at the telemetry system for storing of student pages and archive files.
5. A dedicated commercial desktop computer at the uplink system for command storage and sequencing.
6. At least one router, hub and Internet connection at each school.

Cost for entire hardware system will be around $20,000 the base command and telemetry system. The remaining workstations are standard commercial desktop units based on 1997 technology for Macintosh, Windows, and Unix platforms. Many students may want to have a prime and backup site available for support. This cost does not include the cost of the Internet service provider lines. The command and telemetry lines between the Internet and the end student are variable depending on number of students and acceptable data transfer rates.

1. Allow the RF communication system to lock on to the uplink signal and complete the return link coherently.
2. Stored commanding provides functions currently done in the ground system transferred to on-board functions (i.e., antenna pointing).
3. Telemetry and science data is autonomously downlinked after the return link established. Downlink window is determined by forward link signal strength and the data is stored on board until an uplink command erases the data.
4. Large solid state recorders on-board for storage and playback of the engineering and science data.

Real-Time Operations

An example of one implementation of real-time operations operating with cost effective control center is outlined below. Other implementations are possible and likely, given the unique features of the control center flexibility.

A student residing at George Jenkins High School, Lakeland, Florida, US, has an experiment on-board an orbiting spacecraft whose uplink system resides in a school in Bear Valley at Anchorage, Alaska, US. Prior to the uplink period, as determined from the view period for an orbit tracking program residing on the local workstation, a series of commands are sent to the uplink system over Internet. The commands are encrypted using PGP, transmitted through IP routers configured to allow only specified workstations for commanding. The commands are placed into a command logging area that is protected by username and password and verifies the ordering and format of the commands.

Another student at Schutz & Kanomata - Viavale school in Santa Cruz do Sul, Brazil, with an experiment on the same spacecraft, logs into the same uplink system and places a series of commands on the system. All the commands are viewable, but can modify only the commands that she placed on the system. Once the uplink period starts, the uplink system starts transmitting the relative timed or conditional commands to be sent to the experiment. Any command student can send real-time commands. If multiple command students send real-time commands at the same time, the commands will be buffered and transmitted on a first in/first out basis.

The telemetry is automatically downlinked once it's acquired. The stored science data is downlinked along with the real-time telemetry and repeated to ensure all data is captured on the ground. All the telemetry is transmitted real-time over the Internet in IP blocks and stored for 15 days on a short term archive at the uplink systems site, in this case Bear Valley. Finally, both science and engineering telemetry are transmitted to the student's local workstation.

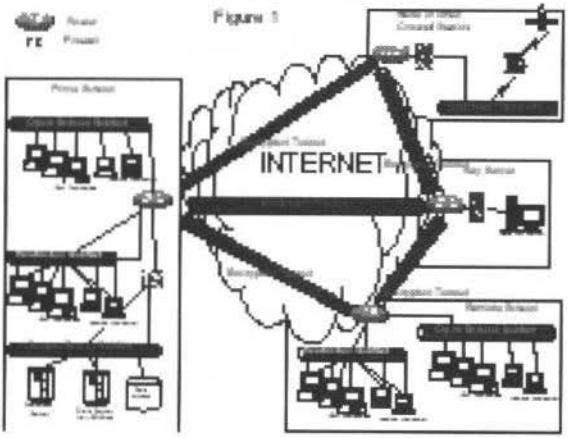


Fig. 1 End-to-End Network Design for Multiple Schools

Summary

Many similar programs address the science and engineering aspects of the space program and forget real-time operations can involve all areas of education. From math and science to language arts can be incorporated into the lessons using real-time spacecraft telemetry.

Also many programs expose the students to space activities concentrate on a particular grade level. The objective of this program is to provide a comprehensive six-week program from 1st to 12th grade (USA). This program is intended to evolve and mature as real classroom instruction begins (estimated fall of 1999). In today's electronic age, flexible, growing, and adapting educational programs can be had with minimal investment. This program will have web pages and electronic distribution of lessons and electronic discussion areas not only for the teachers but for the students involved. Three schools (an elementary, middle, and high school) have been contacted and preparations have just begun to implement these programs in those schools.

Teaching our future engineers, scientists, and leaders at an early age the value of space will provide long-term benefits for all of space and science activities.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp.416-425

ISSN 0100-7386
Printed in Brazil

Truss Structure Tele-operation Using ETS-7 Space Robot

Kohtaro Matsumoto
Sachiko Wakabayashi
Space Technology Research Group
National Aerospace Laboratory
7-44-1, Jindaiji-higashi, Chofu, Tokyo, 182-8522, Japan

Hiroshi Ueno
Tetsuji Yoshida
Space Systems Division
Shimizu Corporation
2-3, Shibaura I-chome, Minato-ku, Tokyo, 105-07, Japan
matumo@nal.go.jp

Abstract

In this paper we describe the results of our truss structure tele-manipulation experiments on the ETS-7 last March 1998. The purpose of our experiment is to establish the basic tele-robotics technologies for on-orbit truss manipulation that was thought as rather sophisticated and also too small size for the current space robots, such as the space shuttle robot arm and the international space station robot arm.

Introduction

ETS-7, was launched Nov. 28, 1997 by NASDA. (Fig. 1) It has been developed to demonstrate two major missions, those are the rendezvous docking and the space robotics. Using the results of ETS-7 rendezvous docking experiment as the on-orbit technology demonstration, the logistic support missions for the space station will be developed as the HTV (H-2 Transfer Vehicle). For the space robotics, Japanese space development baseline scenario request to support the various space activities and/or works for the satellites, the space station, and the moon or planets.

Fig. 1 ETS-7 (Engineering Test Satellite 7)

For the space robotics experiment, four agencies have cooperated with each experiment. Every agency has developed their own experiment apparatuses for the satellite, and their own ground facility for their ground tele-operation experiment. The basic robot systems of the satellite, such as the arm, vision, communication, and the controller, had been developed by NASDA.

Also for the ground system, the basic control and communication facilities had been developed by NASDA. Other three nation institutes have developed satellites components and ground facilities only related to tele-robotic research.

At the initial checkout phase, ETS-7 had reported many trouble of the satellite attitude control system, the high gain communications system for the TV image transfer, and the robot arm assembly. Also the main communication satellite COMETS launched Feb. 1998, had failed to enter the geostationary orbit. In spite of these troubles, the ETS-7 space robot experiments had started from last march, with initial checkout of the robot experiment systems. Because of the COMETS trouble, the ETS-

7 teleoperation could be controlled only in the program mode using the TDRS, until last August. From the last September, the direct teleoperation function was modified for the TDRS communications. Until now, all four agencies have carried their experiment almost one half agencies have carried their experiment almost one half to two third, according to the initial plans.

For our TSE experiment, we also have finished the initial phase, basic phase, using the program mode teleoperation, ETS-7 arm had released the launch lock of the TSE successfully. For the basic phase, we had deployed and stowed the deployable truss structure, and assembled and disassembled the truss connection joint using the program mode control. For the nominal phase, we are running the various innovative teleoperation aid systems. For the advanced phase, from the next march, we will extend our teleoperation freedom and try some VR (Virtual Reality) aid system.

Space Robot and its Ground Tele-Operation

On the space shuttle and on the space station, the robot arms have been used to release satellites, retrieve them, and assemble the space station modules, by the astronauts' on orbit teleoperation. Mainly due to the high level requirements of the space robot on the manned space facility were no implemented or demonstrated with two exception, those are the ROTEX of D-2 and the MFD of STS-85. In the ROTEX, the metal cabinet, that MRF, only the preprogrammed free arm motion was accepted.

These current space robots' status and role seems to be far from the recommended benefits and expectation for the space A&R (Automation and Robotics). For the future space robots, more dexterous tasks and small work objects will be teleoperated effectively from the ground, to release human from the unsuited and/or hazardous works, such as the assembly of large structure and the extravehicular activities.

We have selected the truss structure teleoperation tasks as the model tasks for the future space robots. Due to the diameter of the truss strut and the complexity of the truss joints, the dexterous handling capability of the small target is essential in the truss teleoperation tasks. For the future of the truss structure, it will be used as the essential components of the next generation space station, the solar power satellite, and the space hotel in the far future.

For the ground teleoperation of the space robot on the manned space facility, the teleoperation system shall demonstrate the solution of the two major communication problems; those are the communication time delay and the lack of capacity. Due to the too complex communication system, there is communication time delay more than 5 sec. For the ETS-7, the time delay is also 5,5 sec. For ETS-7, the TV camera images are restricted to 4Hz of JPEG compressed monochrome using 1.5Mbps communication. Because of this low TV image frequency, the robot work site monitoring is restricted only to the hand camera and the shoulder camera. From the viewpoint of space teleoperation, those constraints are essential, such as low quality and low frequency of the monitor TV and long communication time delay. Thus to verify the safety level of the ground teleoperation, technological solutions for those subjects shall be demonstrated through the real space systems' teleoperation.

ETS-7 Robot Arm

The ETS-7 robot arm basic design was introduced from the JEM small fine arm, and slightly modified in its size, harness implementation, and its tool. Its tip position accuracy is quite worse than the ground robot in a factory, because of the vacuum lubrication. For the monitor TV, the arm has a pair of hand eye, and a pair monitor camera on its shoulder, to assure redundancy of the TV monitor. These TV images are compressed into monochrome JPEG at 4 Hz, The endeffector tool has two modes of capturing the grapple fixtures. The finger open operation captures the standard GPF-N that is designed for TSE. (Fig. 5).

The teleoperation has two modes; those are the programs control and the direct teleoperation control. In the program control, every arm tip motion is described as the line motion between two points. before sending the program control command to the ETS-7 arm, NASDA facility checks its motion using the numerical simulator that takes one half or one third of the time required for the real motion, In the direct teleoperation control, the arm tip motion is controlled at 4 Hz directly from NAL's teleoperation facility. On-line verification against the collision and singular attitude, and speed and acceleration checks are required for NAL's facility.

TSE: Truss Structure Experiment

Our TSE (Truss Structure Experiment apparatus) on the ETS-7 is composed from these major parts. (Fig. 2) The TSE was designed and developed to implement the basic truss handling work, those are the

deployment of the deployable truss structure and assembling of the truss connection joint. In addition to these tasks, TSE launch lock was designed to be released by the ETS-7 space robot work.

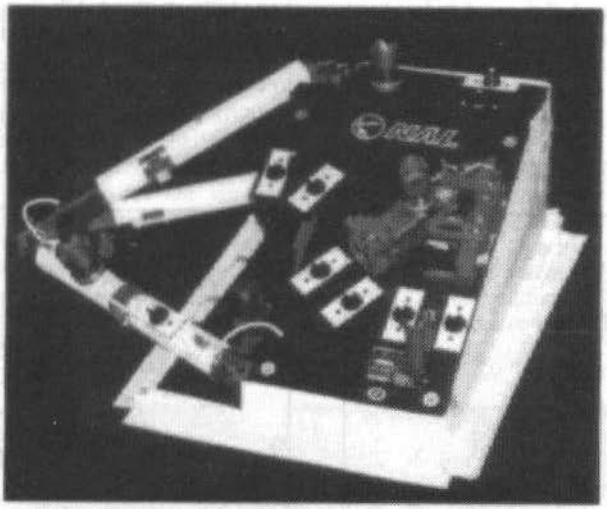


Fig. 2 TSE (Truss Structure Experiment)

Launch Lock (LL):

TSE LL is released by the action of the ETS-7 arm. After capturing the GPF-N (Grapple Fixture-N) of the LL, the arm rotates it with 90 degree, This GPF-N's rotation activates the connected cam' rotation. The cam rotation releases the hook, which locks the sliding blocks. These blocks also lock the deployable truss element and the truss assembly joint. This LL release mechanism is the three-stage domino mechanism, every stages has the mechanical redundancy for its release relying on the arm additional operation.

Deployable Truss Structure (ST):

The latch component of FT is same to the DTB (Deployable Test Bed), that had been proposed and studied for the space station JEM-EF (Exposed Facility). The TSE DT is a part of a triangle truss structure that is statically determinate and can be deployed and folded. (Fig. 3) The DT hinges has 10 DOF (Degree Of Freedom) to handle the on-orbit thermal distortion and the launch shock deformation.

The deploy and stow operation require the precise arm motion along the 3 dimensional spline curve under the closed link condition. And the stow operation requires the latch release by the arm tool rotation and the surmount of the temporal fixture by the arm.

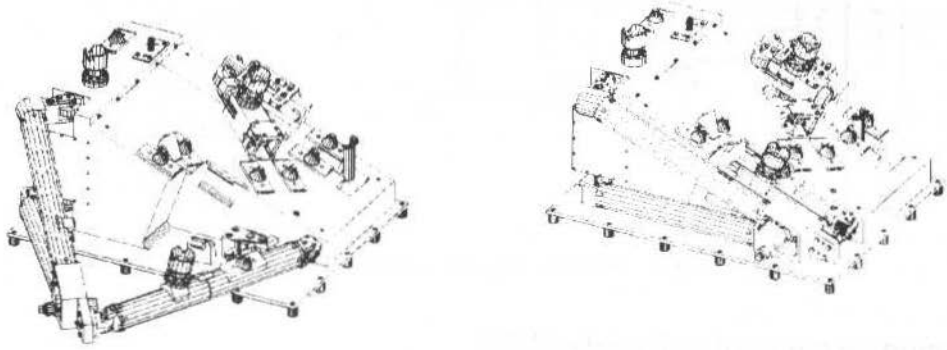


Fig. 3 Deployable Truss Structure

Truss assembly Joint (TJ):

TJ was originally designed for STAR*Bay-2 truss structure designed for human operation. The joint mechanism was modified for one hand robot operation without any hand-over operation. Also the mechanical guides was attached to the joint receiver (JR) to compensate the insufficient accuracy specification of ETS-7 arm tip control.

TJ Assembly tasks are to attach TJ in front of JR, to extend TJ length unlatch operation, to insert TJ into JR, and to latch TJ. Because of the mechanical design constraints of the TJ, the mechanical allowance between TJ and JR is less than 0.5 mm. It is too severe for the ETS-7 arm tip position accuracy, that is ± 2.5 mm as its repeatability specification. (Fig. 4)

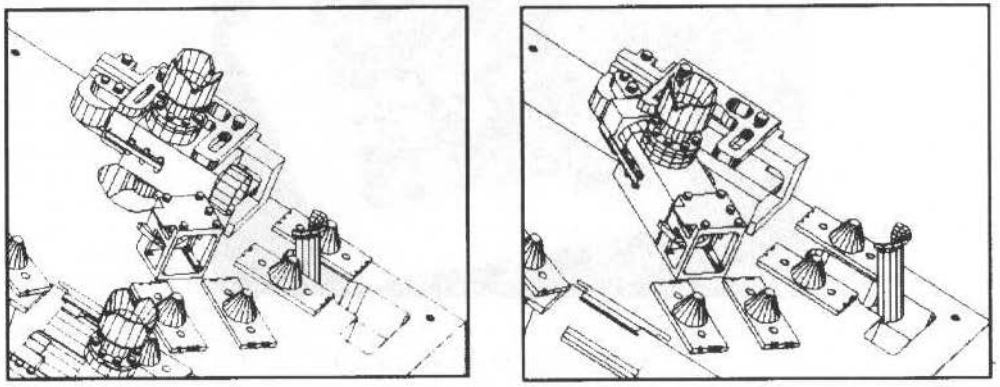


Fig. 4 Truss assembly Joint

Grapple Fixture for NAL (GPF-N):

The ETS-7 robot arm is designed to grasp the three specific fixtures that are GPF-S, GPF-M, and GPF-N. GPF-N is designed as the smallest fixture for TSE truss strut.

The major difference between GPF-N and others are the size and the arm capture method. The diameter of GPF-S&M is 138 mm. The capturing method is also different for GPF-N that uses tool finger close operation, although other fixture uses tool finger open operation. (Fig. 5)

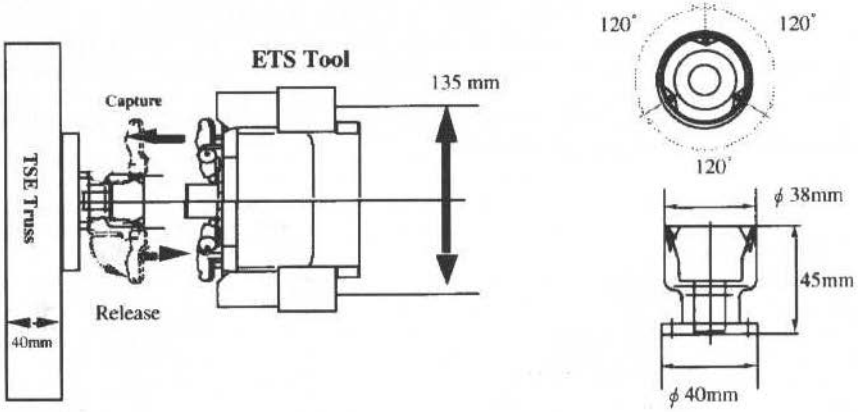


Fig. 5 GPF-N and Finger Close Capture Operation

Ground Teleoperation Facility (GTF)

On the ground we have implemented our tele-operation facility just behind the NASDA's ETS-7 robot experiment facility, The major functions of our teleoperation facility are the followings:

(1) Teleoperation of the ETS-7 robot arm;
(2) Graphic simulation for the predictive display of the robot motion.
(3) Image processing for the precise measurement of the robot arm tip position and attitude.
(4) Hardware simulator of the ETS-7 robot arm with the EM (engineering model) of our TSE.
(5) Joystick for the direct human teleoperation from the ground.
(6) Advanced research software I/F.
(7) I/F to ETS-7 arm through NASDA facility as a transparent monitor.

The hardware simulator is composed of the industrial robot and the engineering model (EM) of TSE, and partially simulates the action and control of the ETS-7 arm. Although the simulator robot is physically different from the ETS-7 arm in its joint assignment and the link length. The tip motion and compliance control characteristics are simulated within the restricted region, where the hardware simulation has the significant meanings, such as the grasping the GPF-N, handling the DT and TJ, and so on. Using this hardware simulator, every experiment procedure and algorithms have been tested and trained before real teleoperation experiment to improve the procedure and/or operation skill under more realistic teleoperation environment than software simulator with graphic. However, because of the physical difference, the results of the hardware simulator do not assure the success of the real ETS-7 teleoperation. (Fig. 6)

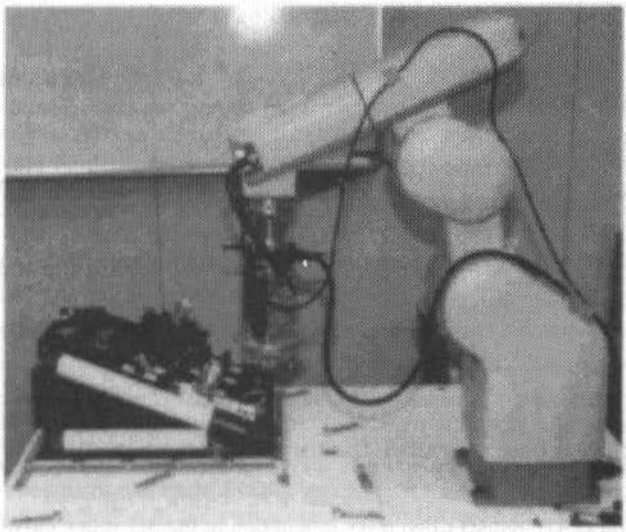


Fig. 6 Hardware Simulator with EM

The advanced research software I/F is developed to enable the innovative telerobotics experiments using the real space robot with minimum and earliest preparation, that is essential to the research purpose. For the teleoperation of the real satellite robot arm subsystem, on-line check and verify process for system safety are essential requirement before sending the teleoperation commands. Those processes usually are the additional tasks for the research, and require much effort to implement them into the research code. The advanced research software I/F utilizes the established and examined safety verifications process in the NAL's standard experiment operation, and accepts teleoperation command from a research code keeping NAL facility's safety reliability. The system safety is also kept by the human operator's override capability that will work as "emergency stop" for any tasks from NAL facility, After the sudden emergency stop, the teleoperation status as same as normal process, and can resume the experiment just as with no emergency.

The communication I/F between NAL and NASDA facilities uses two RS-232C lines with 9600 bps. One RS232C line is used for teleoperation command and others for the telemetry data distribution. using the RS232C, both facilities can avoid the network and system trouble, and get more flexibility and adaptability. The teleoperation command is generated at NAL facility at 4 Hz and is send to the ETS-7 through NASDA. NASDA facility works as a transparent communication line with safety check monitor function, from the viewpoint of the robot teleoperation between NAL and ETS-7 robot. The telemetry comes at 10 Hz. JPEG compressed monitor image are decoded and distributed as NTSC images (Fig. 7)

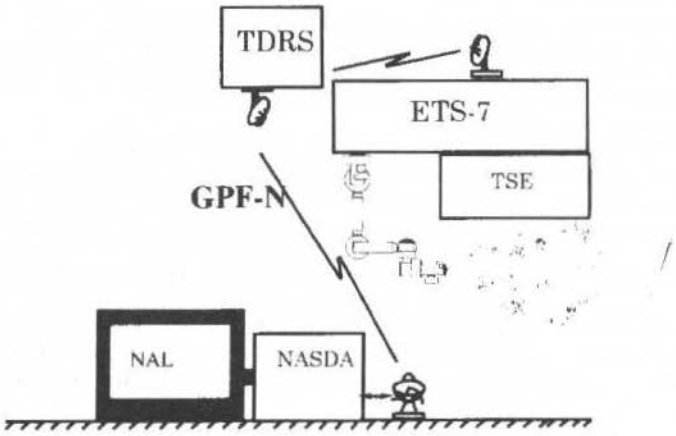


Fig. 7 NAL-NASDA I/F

TSE Experiment Tasks

TSE experiments have three tasks for the ETS-7 robot teleoperation, those are 1) Launch lock (LL) release, Handling the deployable truss (DT), and 3) Handling the truss joint (TJ). Every task requires GPF-N capturing process by the ETS-7 arm. TSE has three GPF-Ns for LL, DT, and TJ.

LL release:

The arm rotates LL's GPF-n 90 degree by tool rotation. The major difficulty of this task is the GPF-N capture itself, since the capture task is the first operation for NAL, without any trial or experience at the ground.

Deploy DT:

The arm deploys DT long a 3 dimensional spline curve under closed link arm control, The technical difficulty is to move the arm along the trajectory within suitable tip force and torque. The closed link movement along a strictly defined trajectory is the first operation for ETS-7.

Stow DT:

The stow operation of DT requires two tasks in addition to the 3D spline curve movement. The first one is lock release by the GPF-N rotation with 25 degree. At the final stage of stow process, the arm is requested to surmount the temporal fixture. In addition to the truss stow operation, the GPF-N capture task is also difficult, since the location and attitude of the GPF-N at the truss deployed state is less stable than other GPF-N due to the looseness of the truss hinges.

Assemble TJ:

The TJ assemble operation has four stages , to swing TJ to the front of joint receiver (JR), to rotate GPF-N 60 degree to unlock and extend the TJ top, to insert TJ into RJ, and to lock TJ by rotation. The TJ insertion operation is the most difficult operation for the ETS-7 arm, because of the too severe mechanical allowance between TJ and RJ. During the insertion process, the teleoperation system has to control the ETS-7 arm in the most precise movement with strong force, that is vertical to the tool axis.

Disassemble TJ:

This tasks is done according to the reverse order of the assembles task. Since the TJ has a spring to stow TJ automatically, only one difficult task is to extract TJ from RJ.

TSE Teleoperation Experiments

Until the December 1998, we have done the following experiments gradually, from the launch lock release to the joystick teleoperation, using 12 mission days. In our all experiments, the responsibility for the safety and the mission certainty by our teleoperation is the most critical factor and are to be checked

attentively. The teleoperation safety of these SOPs was pre-examined and real-timely examined by our graphic simulator. The teleoperation certainty of these SOPs was pre-examined by the hardware simulator repeatedly during the training and rehearsal.

For the arm tip control, the rigid position control is used during the work site access phase and the fine running phase before the GPF-N capturing. From the final capturing process and all over the truss handling tasks, the soft or rigid compliance control, or the active limp control, is activated to avoid excessive force from the arm.

LL Release:

The LL was released in the last May, by the program control teleoperation mode. After capturing the GPF-N, there was no trouble for this operation. The three-stage domino mechanism worked very well to release both launch locks simultaneously. Fortunately, the redundancy mechanism for LL was never used.

Program Control Teleoperation:

The deploy/Stow of DT and assemble/disassemble of the TJ were done from May to July, In the program control mode, the basic arm tip trajectory and teleoperation sequences were pre-programmed into SOP (Sequence Of Procedure) using straight-line movement between two points. The design data and slight modifications from the hardware simulator results were used as the base of the preprogrammed SOP. Also a small set of minute adjustment programs of one axis movement were prepared for the critical phase and for the final phase of the tasks.

The Program Control of DT:

During the DT handling, piece wise straight lines approximation of the 3D spline curve accomplished the deploying and stowing tasks within a pre-estimated force and torque, except the final stage of deployment and stow. At the final stage of deployment, because the trajectory needs relatively steep curve, and the estimation error of the TSE installation to the satellite become large, unexpected large force and torque was observed, and sustained within acceptable range using the prepared minute adjustment programs. Also at the final stage of stow, the resistance force from the temporal fixture was different from the hardware simulator because of the difference of the gravity. For this phase operation, the prepared minute adjustment program was also used to finish the stow task.

Program Control of TJ:

During the TJ handling, piece wise straight-line trajectory was used for the swing operation. The unlock operation was programmed as a simple tool rotation, and executed. But at the TJ insertion phase, the miss-match between the mechanical guide and the ETS-7 arm control accuracy became the great barrier of the insertion task. The preprogrammed straight-line command could not insert the TJ into its receiver RJ. The prepared minute adjustment program also could not deal with this barrier. Before the success of TJ three mission paths and trials. Finally we changed the sequence of teleoperation process, to avoid the effect of the tool rotation to the swing operation in the program control mode teleoperation. Since this coupling effect is closely related to the arm control system in the program control mode, we could not identify of neglect this effect during the hardware simulator examination. The tentative conclusion for this problem is that for this task we needed more detailed dynamical analysis of the relation and arm behavior from the position hold stage to the beginning of straight-line movement. The position hold stage is automatically taken between a pair of straight-line movements.

For the disassembling of TJ, only one concern for the program mode teleoperation is the arm tip force. During the disassembling task, there was no problem.

Direct teleoperation Program Control:

From September, the direct teleoperation control has been enabled, using TDRS. As the direct teleoperation program control operation, that is a preprogrammed direct teleoperation control, using model based estimated trajectory. Every trajectory of DT and TJ were calculated using inner truss model. This truss model is based on the design data and can calculate the trajectory with the finest precision. In this control mode, every arm movement was directly controlled from the ground, and thus we could fully control the on-board arm at 4 Hz. The arm tip force becomes more controllable than the program mode control. Instead of the minute adjustment programs, the joystick is used for the fine-tuning during the critical phase.

Direct Teleoperation of DT:

The arm is controlled precisely the calculated 3D-spline curve. If the required DT handling force is negligible small, the force during this task shall be so small. However, because of the spring force at the

hinges, DT handling requires some handling force. The arm position control just on the precise spline curve, does not mean the optimum trajectory. And the TSE installation error problem still remains. Thus the force observed during the DT deploying and stowing is almost same to the force during the program controlled DT handling. (Fig. 8) The problems of the steep curve at the final deployment phase and the difference of the resistance force from the temporal fixture are also observed, and handled by the joystick operation.

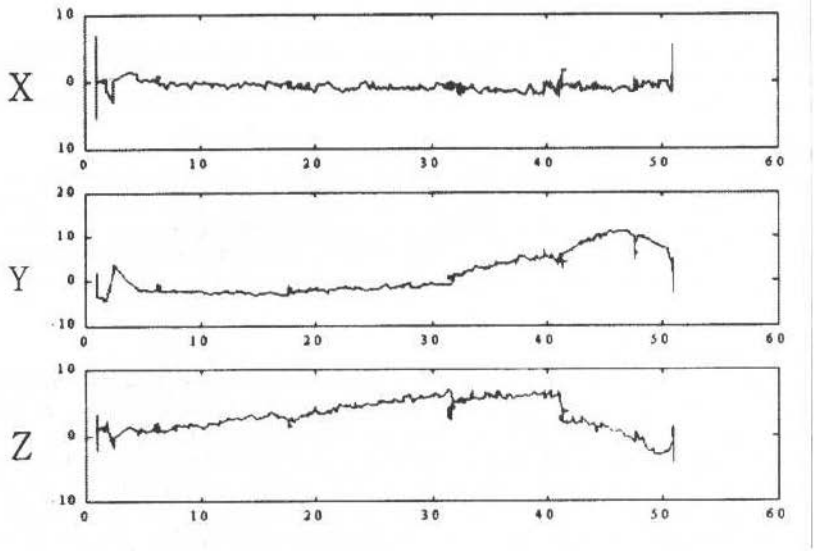


Fig. 8 Force during Truss Deployment

Direct Teleoperation of TJ:

The benefit of the direct teleoperation appears well in the TJ handling. For the swing of TJ, almost no difference observed. But no position holds stage during the operation becomes the advantage at the TJ insertion phase. During the insertion, when the top of TJ stacks at the entrance of RJ, the direct teleoperation program can continue from that stage linearly, and handle TJ with shake or jog to meet the too severe mechanical allowance just like the human operation. Therefore in the TJ insertion, the joystick fine tuning operation works as the major phase even in the direct teleoperation program. The operator made many try and error operation to seek the right entrance position. These try and error teleoperation might be essential for the tasks y the less accurate arm, just like the ETS-7.

Overall and Future Experiment Plan

TSE experiment is assigned 28 mission days / 56 mission paths during the ETS-7 official mission period. In each mission path, the real robot experiment can consume 20 to 30 minutes. The 20 minutes is the baseline mission time. The excessive 10 minutes is the margin for anomaly phase of the experiments, such as the failure of the image processing measurement, unexpected strong force during operation, and so on.

NAL's 28 mission days are classified as follows;

1) Initial checkout (2 days): Release LL, ...
2) Basic Experiments (8 days): DT and TJ experiments using the program mode control and the direct teleoperation control.
3) Nominal Experiments (11 days): Various teleoperation aid system.
4) Advanced Experiments (7 days): Virtual reality, Long communication delay more than 10sec.

From the last December, we began the nominal experiments, such as the unknown trajectory teleoperation and force feedback teleoperation.

The mission life of ETS-7 will continue to the end of next May: Until that time, we will carry the nominal and advanced experiments 3 mission days per month. For the future space robotics, our experiment subjects will be focused into the next generation teleoperation support technology, to realize

the real ground teleoperation for the station and its successors, and to release human from the hazardous tasks and/or routine tasks.

Concluding Remarks

The Truss Structure Experiments (TSE) by the ETS-7 robot arm has been done to demonstrate the space robot capability for the truss structure construction work, that requires rather complex dexterity and is small work objects. The ground facility for the TSE experiments has been used and shown its reliability for the space robot teleoperation. Through the TSE experiments, the advantage and disadvantage of the program mode control and the direct teleoperation control were demonstrated.

For the future of the space robot teleoperation, we will continue the experiments to develop the advanced teleoperation support technology, using the graphic simulation, force feedback joystick, and other innovative aid systems until next May.

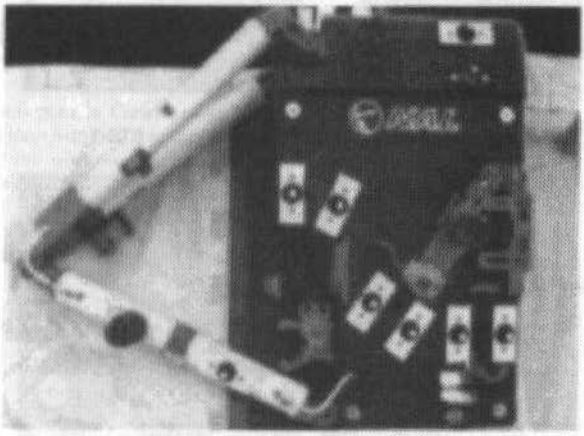


Fig. 9 Deployed & Assembled TSE

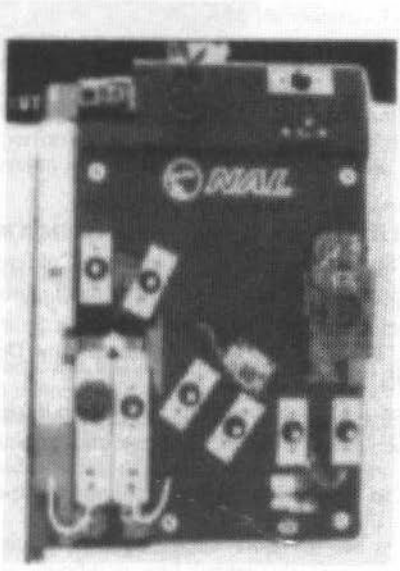


Fig. 10 Stowed & Disassembled TSE

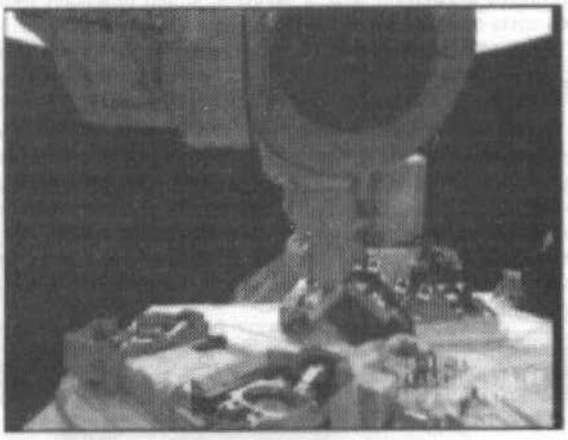


Fig. 11 Truss Deploy Teleoperation by ETS-7

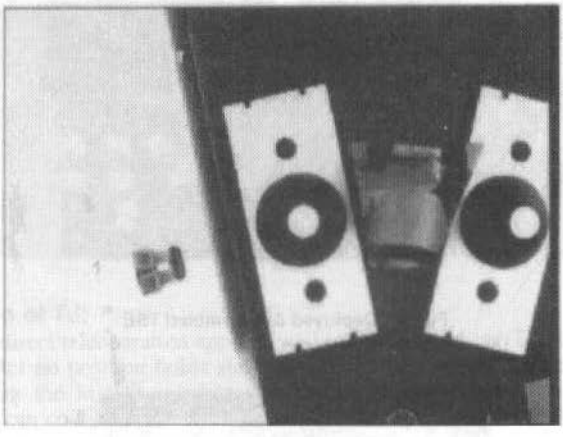


Fig. 12 Hand Eye TV Image for Deployable Truss

References

H., Ueno; T. Yoshida, S. Kibe, and K. Matsumoto, 1993, "Study on Joint Mechanism for Truss Structure", Proceedings of 37th Space Sciences and Technology Conference Kitakyushu, (in Japanese)

K. Matsumoto, et al., "Truss Structure Tele-Manipulation Experiment using ETS-7", Proc. 3rd ISAIRAS-94, Pasadena, 1994

S. Wakabayashi & K. Matsumoto, "Teleoperation with Additional Time Delay Mechanism for ETS-7 and JEM-EF", 4th IAF, Beijing, 1996

K. Matsumoto, et al., "Development of Truss Structure Teleoperation Experiment by ETS-7 Robot", Proc. ISAIRAS-97, Tokio, 1997

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp.426-433

ISSN 0100-7386
Printed in Brazil

SACSO a Computerized Engineering Workshop for Systems Simulation

Antoine de Framond
Pierre Chauchat
SEMA GROUP
Innoparc, 1 rue de la découverte
B.P. 46, 31675 LABEGE Cédex France
adf@labege.sema.fr

Jean-Louis Dulot
CNES
18, avenue Edouard Belin
31401 Toulouse Cédex 4, France
jean-louis.dulot@cnes.fr

Abstract

*SACSO (*Structure d'Accueil pour la Construction de Simulateurs et leur mise en Oeuvre*), is a UNIX-environment software engineering workshop bringing together all the activities related to the numerical systems simulation.*
Some of the highlights of the workshop include facilitation of exchanges between users, of the elements they are developing whilst respecting their autonomy, easy resources (personal and joint) accessibility, and re-usability due to standardized interfaces.
The software architecture and the originality of the solutions used means that SACSO is not necessarily dedicated to a particular professional branch and so can be used as a generic solution for other systems.
Further developments are planed to the current workshop to enhance user-friendliness and to increase the present range of applications.
Keywords *:Simulation, Dynamic systems, Numerical integration*

Background

The Dynamics and Automation department of CNES carries out numerous studies and performs expert appraisals prior to projects, during projects and as part of R&T, in the area of satellite and payload attitude control. Simulation is a quick, low-cost way of validating the suitability of an Attitude and Orbit Control System (AOCS) for a particular mission, of checking that the operating range for each component has been respected, of adjusting and studying the behavior of a flight control loop (acquisition after a change of AOCS mode, steady state, response to perturbations), of studying dynamic coupling (AOCS/payload, AOCS/solar array), of estimating performance based on statistics, of reproducing a phenomenon observed during system operation in order to better explain it, etc.

The SACSO project was given the go-ahead in early 1996 in keeping with CNES's determination to increase the efficiency of work performed by engineers. SEMA GROUP began development activities in April 1997, and a preliminary version was delivered in December 1998.

Role of the SACSO workshop

Based on the observation that increased efficiency in the realm of simulation requires harmonizing working methods, perpetuating know-how and focusing on the professional branch rather than on computing tools, five basic functions were assigned to the SACSO computerized engineering workshop:

1. To manage the department's technical heritage (models and simulation results common to the team or peculiar to each member);
2. To help develop simulators using a graphic interface and centralized numerical integration, the only way of strictly taking into account the interdependence of the states of the various components in a dynamic system;
3. To develop a scenario-programming interface (definition of simulation sequences and their operating context) ;
4. To enable interactive control of applications (stopping and restarting a simulation, "real-time" modification of simulation conditions and dynamic display options);
5. O exploit the results using an interface with data processing packages such as MATLAB.

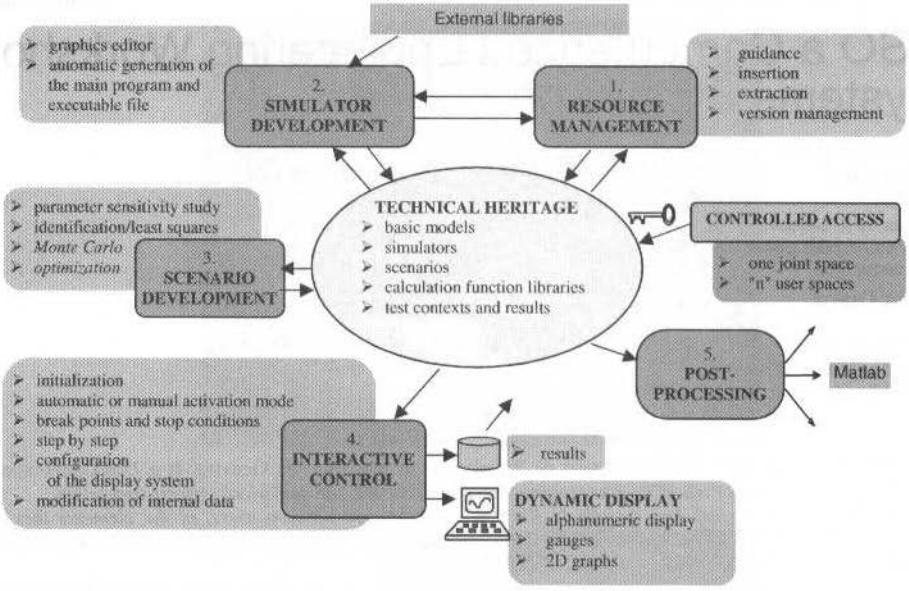


Fig. 1 Functional description of the SACSO workshop

Advantages of the SACSO Workshop

A joint Working Platform for the Whole Team

SACSO is a multiple-user host structure which unites all the activities related to the simulation of dynamic systems. The workshop facilitates exchanges between users of the elements they are developing while respecting their autonomy. Each user has his own environment (with his own personal resources) and can draw upon the technical heritage of the whole department (joint resources).

Guaranteeing Future Relevance and Facilitating the re-use of Models

Personal resources can be configuration-managed in the same way as joint resources. The person in charge of joint resources ensures that each module is well-documented and validated. Joint resources are directly accessible to any declared workshop user. Re-use is also made easier by standardized interfaces.

Keeping Simulation Models Under Control

A user designs a simulator (orbiting satellite with an AOCS, instrument and its pointing system etc.) by assembling elementary models (space vehicle or instrument dynamics, attitude sensors, measurement filters, actuators, control laws, orbital environment etc.). The models are encoded either by the user himself or by other users in C, FORTRAN 77, FORTRAN 90, C++ or ADA95. He retains full control of both the software code for his models and the design of the complete simulator.

Remaining Open to the Outside

SACSO can accept widely differing models due to the simplicity of the interface standard, which can be adapted to any language. The workshop also allows the user to employ modules from external libraries (NAG, IMSL, MSLIB, etc.) in the simulators and basic models. The preliminary version does not include the exportation of simulators outside the workshop, but this feature will be integrated in a subsequent version. Export functions are used to convert results files into different formats, including a MATLAB format and an ASCII format easy to exploit using any other post-processing software package.

Controlling Simulation Runs

The studies and expert appraisals carried out by the department require great flexibility in simulation implementation. The workshop enables simulations to be run one after the other automatically by defining scenarios, though the user can intervene at any time using the interactive GUI (Graphical User

Interface). The dynamic display of results (i.e. graphs which change as the simulation progresses) facilitates simulation control and monitoring.

Within a scenario, the algorithms for a scenario may be programmed. These algorithms may vary from the study of parameter sensitivity and the identification of parameters by the least squares method to optimization by a gradient technique and stochastic analysis using a Monte Carlo method etc.

The user controls simulations using a control panel with which he can insert break points and stop conditions, suspend or stop the current simulation, modify internal data or the way data is displayed while the simulation is ongoing, move to step by step mode (each step being defined by a whole number of integration cycles) or to debug mode, when the workshop calls upon the operating system's debugger.

Concentrate on Professional Aspects... Not the Ups and Downs of Computing

Insofar as possible, the workshop deals with all the recurrent or computerized aspects of development, including the implementation and management of simulators, which is not directly linked to the job description of engineers in the Dynamics and Automation department (who are principally concerned with general mechanical engineering or automatic features).

Basic Concepts

Basic Models, Global Models and the Simulator

A *basic model* is an elementary part of a simulator. It includes a source code and various types of interface data (configuration parameters, continuous or discrete state vector, inputs and outputs), which are not necessarily always required.

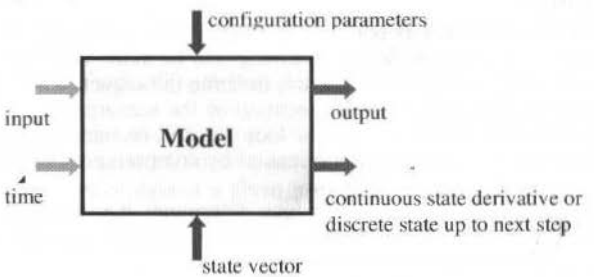


Fig. 2 Interfaces of a basic model

A *global model* is obtained by assembling several basic models (connecting outputs to inputs). A global model can be made not only of basic models but also other global models. The simulator is the highest level global model.

SACSO integrates three types of model:

- Continuous models with a continuous output;
- Continuous models with a sampled output;
- Discrete models.

Each type of model can include states or not.

There are different activation phases:

- Internal initialization,
- External initialization,
- Computation of derivatives (for continuous models) or the state (for discrete models),
- Computation of outputs,
- Saving the simulation context,
- Restoring the simulation context.

The initialization phases of models consist of initializing internal variables and outputs so that the models benefit from correct inputs at the beginning of the integration cycle when the inputs are connected to the outputs of other models.

When there is a loop, the input of the first model executed is not initialized at the end of the first initialization phase. This is why two sub-phases are distinguished:

- Internal initialization phase: the model only computes internal variables and the independent outputs of inputs,

- External initialization phase: only internal variables and those outputs directly or indirectly dependent upon inputs are computed,

after which the external initialization phase is iterated.

SACSO lets the user choose the number of external initialization cycles to be completed.

Simulator Design Using a Graphics Editor

SACSO includes a graphics editor for use in simulator design (inherited from the LRBA's OSAACAS basic product, cf. § *computer architecture*). The editor uses the schematic representation of a model shown in figure 2. Connections are made using the mouse. Both new models and models from the workshop's *technical resources* (personal or joint resources) may be inserted. For new models, once the model interfaces have been defined, a skeletal code is generated automatically. The user then inserts his own algorithmic code into the skeleton. When the model has been completed, SACSO generates the code for the simulation monitor and creates an executable file.

Centralized Numerical Integration

The simulators created using SACSO are not "event-oriented", in which case each model controls its own state. Simulators of this type are unsatisfactory when modeling a complex dynamic system such as an orbiting satellite because they cannot handle all the couplings between the various system components. In SACSO, on the other hand, the state variables for all the basic models are gathered together into a global state vector and it is the simulation model which handles calls to both the numerical integrator and the various models. Different numerical integrators are available in SACSO: Euler (order 1), Runge-Kutta (orders 2, 3 and 4), and Adams-Moulton (order 4).

Scenario and Algorithm

The conditions in which a simulator is operated are defined by a *scenario*. SACSO can be used to create scenarii of a greater or lesser complexity involving one or more simulators. In its more general form, a scenario can run several simulators in sequence, iterating the sequence at will.

The algorithm is a method encapsulating the execution of the scenario which permits to manipulate the configuration parameters. For example iteration loop on one or more parameters to examine the sensitivity of simulated system or parameters identification by comparing simulation results with sample measures.

The iterations are controlled by an *algorithm* which determines the context of initial data for each simulator and controls the end of iteration depending on the calculation of the stop criteria.

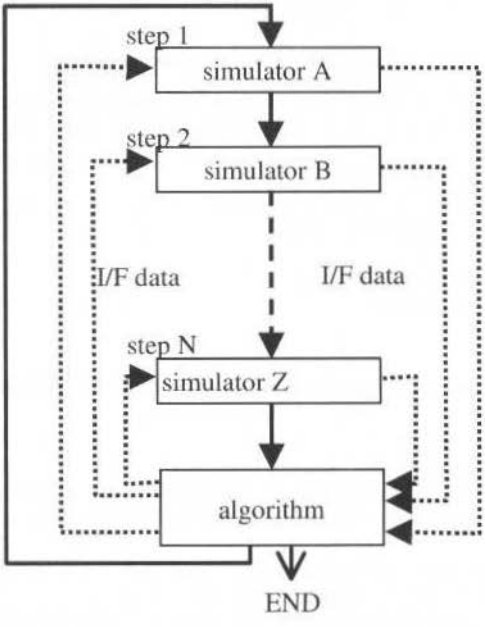


Fig. 3 Principle of scenario implementation

Object modeling

The simulation components are seen like objects that the user can put together to design the simulation. The main types of objects are : models, simulators, scenarii and algorithms.

The model is the smallest entity of the workshop in which the user implements his calculation function. The same model can be instantiated several times and inherits the same calculation function. The advantages of which include non duplication of source code and will allow a unique model to represent a physical element which can be used several times (eg : a three axis gyrometer is composed a three times a one axis gyrometer).

The simulator is a set of models, connected together. It is a inherited class of model which contains a lot of models. The user can reuse previously made models in other simulators.

The scenario is composed of one or more simulators linked together. It also defines the initial values of configuration parameters and states vectors.

There are four standard algorithms in SACSO, but the users can implement their own algorithms and put it on the department patrimony like the other types of objects.

All these components are class instances, and objects composition is very largely used in the simulation realization. In fact a "simulation" is a scenario execution with an algorithm, it can be composed of more than one simulator.

This object modelling is practical for the user because he finds the same concepts and operations in the different stages of his work.

Objects management is similar in the whole GUI (Graphical User Interface). It is an important way of increasing ergonomics and facilitates workshop learning.

Resource Management

The elements generated by or integrated in the SACSO workshop and which constitute the department's technical heritage, are as follows:

- Models, simulators and scenarii with their default initialization context (input data and parameter configuration data),
- Basic functions for numerical calculations,
- Test contexts (simulation results, associated initialization contexts and messages),
- Simulation contexts (set of data at a given instant during simulation, used to run the simulation again at a later date beginning with its state at that particular instant),
- Post-processing functions and results.

These resources are divided up into *joint resources* and *personal resources*. Each user owns his own personal resources to which he alone has access (SACSO thus respects the industrial confidentiality surrounding certain projects). Joint resources are managed by the workshop *administrator*, alone authorized to add to or modify the resources using elements from personal resources sent to him by the users.

Users may thus:

- Develop simulators and define scenarios using not only their own personal resources but also those of the whole department,
- Swap elements (models, simulators, scenario algorithm etc.) with other users,
- Enable the whole team to benefit from their own work and developments by sending items to the workshop administrator.

All resources except text contexts and simulation contexts may be archived and configuration-managed (by incrementing the version under RCS)

All the object types handled in a simulation can be archived in the personal or department patrimony. When the user archives an object, all the files which are part of this object are archived by the workshop with a version identification. It also takes care of the versions of other objects which refer to this one. When the users want to reuse a version of a scenario, the workshop extracts automatically the corresponding versions of the simulators and models.

The users access' the resources by a navigator which is generic to the whole workshop whatever the type of component used.

Computer Architecture

SACSO is built around the LRBA's OSAACAS software. OSAACAS is a tool for developing missile simulators based on the assembly of elementary models using a graphics editor.

All the functions of OSAACAS are kept in SACSO. The most important of these is the hybrid real time simulations. A simulation not only contains numerical models, but it can also contains Real Element Interface (REI) in which case it is a hybrid simulation.

REI can be used to test a part of a dynamic system with existing elements (numerical or analogical). The numerical models simulate the rest of the system, for example the environment. The real elements can be easily inserted into the simulator like other models.

An REI is like a model except that its connections describe data flows. The simulation monitor takes into account the real time synchronization between models and data flows. The REI are small tasks executed on processor cards on a VME rack driven by the real time operating system VxWorks.

There are two modes of synchronization: real time and pseudo-real time. In real time mode, the models and data flows are synchronized by a hardware clock which controls the start of the model calculation and data transfers. In pseudo real time mode, the synchronization is driven by the end of model calculation and the end of data transfers.

To dispatch the calculation load, the models can be executed on several processors. If two models on different processors are dependant, the simulation monitor waits for the end of the first model before starting the second model.

The simulation performance can be verified even without a real-time execution: in fact a satellite simulator used for the numerical validation executes as rapidly within the SACSO workshop than the previously coded simulator.

The modularity of REI implementation allows a great flexibility in hybrid simulation. In fact you can add new real elements by replacing numerical models. This operation is facilitate because they have the same standardized interface in the workshop. If the number of real elements grows importantly, you can add new VME cards and simply specify it in the configuration files.

Software Architecture

The workshop is based on several tools. The most important software are Oracle for the persistence of objects, RCS for the management of resources, Ilog Views and Inform for the design of GUI. SACSO built around three general entities : Conception (composed of simulation conception supported by resources management), Execution (supported by scenario definition) and Services.

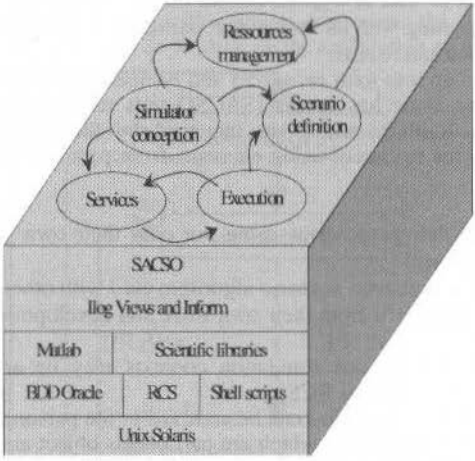


Fig. 4 Software layers

Oracle is used for handling the list of objects and their characteristics in conception phase. It allows a great flexibility in handling the links between the objects. It is easier to change the connections between models or add a model in a simulator.

Database is handled in several ways: by database triggers, stored procedures, SQL scripts and ProC subroutines. Database triggers check the validity of user operations and take care of objects consistency.

The SEMA GROUP development team created a service library which manages the object files list and the links between them. So a version of an object which references versions of other objects can be automatically saved or extracted using RCS.

The service library is also used by all the other application tasks. There is for example a centralized messages log. Process management is also improved by this library. The application checks that certain programs are not already in action. It also kills the process hierarchy when the user wants to leave.

To improve performance and flexibility, the workshop produces an independent binary program and simple ASCII files for the parameters. The simulation models can be written in some languages like F77, F90, C, C++ and ADA95. The simulator monitoring kernel is constant and written in C language. It drives simulation stages and uses the models calling subroutine which is automatically generated by SACSO. So the workshop compiles and links models, calling subroutine; simulator kernel and eventually external scientific libraries to produce an executable file which can runs alone or can be controls by the execution GUI.

SACSO Development

Development began with a definition phase lasting 5 months. This included the writing of software specifications and preliminary design. This phase involved an GUI mock-up developed in close cooperation between the developer (SEMA GROUP) and the customer (CNES). The iterative design of the mock-up allowed the GUI ergonomics to be modified and adapted according to the customer requirements.

The conception phase lasted 3 months. The OMT method was used during the conception. It allowed the object oriented design to define the classes and dynamic exchanges.

Reviews were held at the end of each phase in order to encourage a closer working relationship between the developer and customer.

The development phases follow a V cycle.

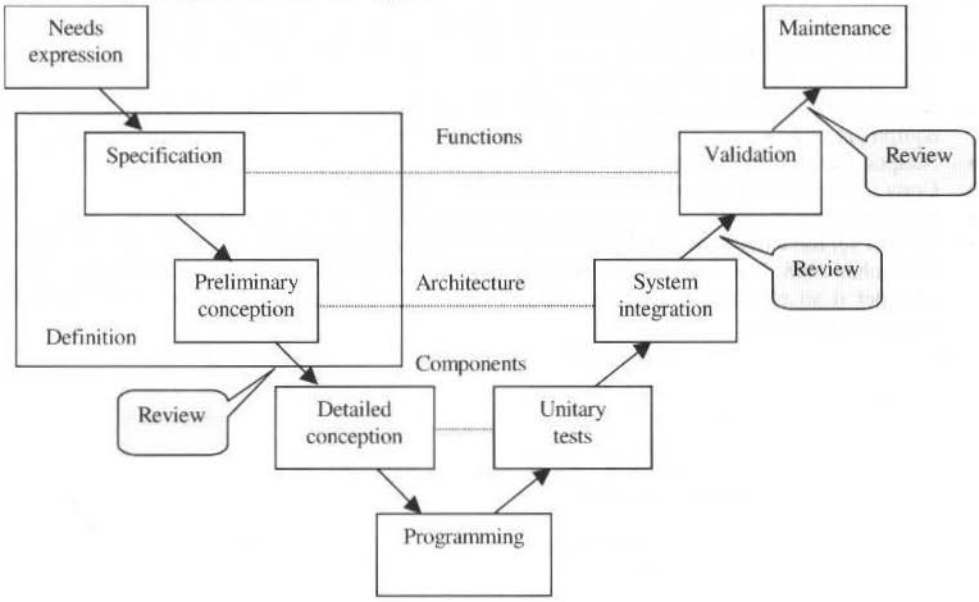


Fig. 5 Development cycle

The test phase was particularly challenging with a functional validation as well as a numerical validation. For the numerical validation some elementary simulators were used to test the state integration for all the methods implemented in the workshop. A complete satellite simulator (Proteus satellite AOCS) was also converted in SACSO models (40 models) to validate results which were successfully compared with original results

Quality checking was applied during the project. In additon to the customer requirements, the SEMA GROUP's internal system quality (ISO 9001 compliant) was rigourously followed and successfully audited on numerous occasion. The quality rules were elaborated in collaboration with the customer. The CONCERTO AUDIT software product developed by SEMA GROUP was used to check automatically the program sources during the realization phase. It allowed easy and continuous checking of programs developed by the team.

The multidisciplinary development team was composed of eight members. The integration of the numerous software tools required specialists in various domains (database, GUI, object programming, simulation, numerical methods, etc...).

Finally SEMA GROUP gave a training to all the future users of SACSO workshop. It provided them with all the know-how to develop simulation with efficiency.

Conclusion

SACSO is a very complete computerized simulation workshop as it covers all the activities relating to studies of dynamic systems involving simulation: from managing the technical heritage of a department and exploiting the results obtained to developing and implementing simulators.

SACSO is not dedicated to one particular professional branch. It may be used for applications other than an AOCS or a satellite, for example :

- Images processing,
- Data flow processing,
- Non dynamic systems.

Further developments are planned to enhance user-friendliness and increase still more its wide range of applications:

- Simplification of the GUI and internal resource management mechanisms so as to optimize response time during simulation design,
- Inclusion of new numerical integrators, particularly variable-step integrators,
- Writing variable-step data to files,
- Integration of event-oriented mechanisms: acyclic activation of modules, occasional data input (downloaded data), run instructions in the form of a sequence of commands with conditions on the state of simulation (change in variable value, modification of integration frequency, transmission of a message to the log etc.),
- New hybrid simulations with a less performance hardware configuration to improve satellite equipment testing such as star sensor, multi-axis gyrometer, solar array driving mechanism,
- Adaptation to other sectors within CNES,
- Copy / paste elements from one graphical design to another,
- New functions to improve optimization and identification of simulation parameters.
- IT is as yet too early to assess the use of SACSO, as the preliminary version was only delivered in December 1998. However, engineers in the CNES Dynamics and Automation department already consider it an efficient tool which will not only facilitate their work but will offer them new analysis capabilities.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 434

ISSN 0100-7386
Printed in Brazil

Posters

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 435-448

ISSN 0100-7386
Printed in Brazil

A Linear Solution for the Artificial Satellite's Attitude Optimal Control

Regina Kuranaga dos Santos
Maria Cecília Zanardi
Group of Orbital Dynamic and Planetology
Department of Mathematics FE UNESP
12500-000 Guaratinguetá, SP Brazil
cecilia@feg.unesp.br

Sandro da Silva Fernandes
Department of Mathematics
Instituto Tecnológico de Aeronáutica
12228-900 São José dos Campos, SP Brazil

Abstract

A first order analytical model for a general problem of artificial satellite's attitude corrections maneuvers submitted to gravity gradient torque is presented in this paper. It is assumed that the satellite is a rigid body, with cylindrical or spherical symmetry and its orbit can be elliptical or circular. The problem of optimization is formulated as a Mayer problem and the control torques are provided by a power limited propulsion system. The state is defined by Andoyer's variables and the control by the components of non-conservative external torques in the artificial satellite's axes of inertia. The Pontryagin Maximum Principle is applied to the problem and the optimal torques are given explicitly in Andoyer's variables and their adjoints. The problem of optimal attitude corrections given by the linearized Hamiltonian around the reference attitude is also analyzed, considering the mean Hamiltonian related with the gravity gradient torque. The complete first-order analytical solutions for the problem with fixed duration are gotten by simple quadratures. A law of the optimal control is proposed and the required optimal consumption is presented.
Keywords: *Attitude corrections, minimal fuel consumption, Mayer problem, Andoyer's variables.*

Introduction

The satellite's attitude represents how the satellite is oriented in space. The attitude expresses a relation between two coordinate systems and can be represented by the Euler angles. In this paper, we assumed a 3-1-3 sequence of three consecutive rotations about the satellite axes for the Euler angles (φ , θ , ψ), to define the relation between the system of the artificial satellite's principal axes of inertia (Oxyz) and the system OXYZ (with axes parallel to the axes of the Earth equatorial system).

The dynamic system associated to the satellite's attitude maneuvers will be expressed in terms of the Andoyer's variables (Zanardi, 1986) (Deprit, 1970) $\ell_i , L_i , i = 1,2,3$, shown in the Fig. 1. The angular variables $\ell_i , i = 1,2,3$ are angles, which are related to the coordinate systems Oxyz and OXYZ. The metric variables are defined as: L_2 is the modulus of the total angular momentum, L_1 and L_3 are respectively the projection of L_2 on the z-axis's principal axis system of inertia and on the inertial Z-axis.

The transformation between Andoyer's variables and the Euler angles is well defined (Zanardi, 1986) (Zanardi, et al., 1994), using properties of spherical trigonometric associated with the spherical triangle $\mathbf{N_1N_2N_3}$, shown in the Fig. 2. It is possible to prove that this transformation is canonical (Deprit, 1970).

The rotational motion of the artificial satellite with cylindrical symmetry, considering the non-conservative external torques and Andoyer's variables, can be expressed by (Zanardi, et al., 1994):

$$\frac{d\ell_i}{dt} = \frac{H}{L_i} + P_i$$

$$\frac{dL_i}{dt} = -\frac{H}{\ell_i} + S_i \quad , \qquad (1)$$

i= 1,2,3 ,

where:

$$H = \frac{1}{2A} L_2^2 + \frac{1}{2}\left[\frac{1}{C} - \frac{1}{A}\right] L_3^2 \ , \qquad (2)$$

where A and C are the satellite's moments of inertia on the axis Ox and Oz, respectively, and P_i and S_i depend on the components of non-conservative external torque (Zanardi, et al., 1994) in the system Oxyz.

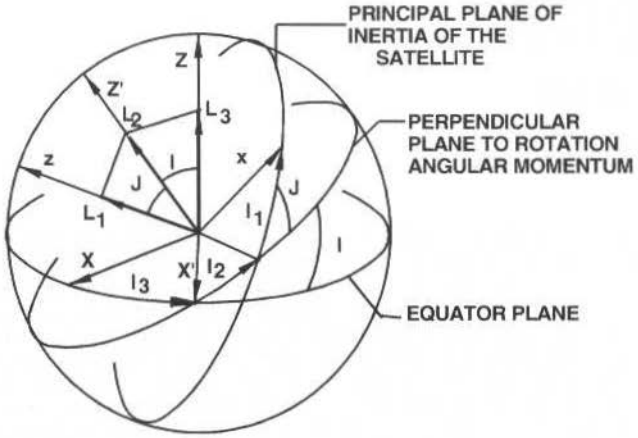


Fig. 1 Andoyer's Variables

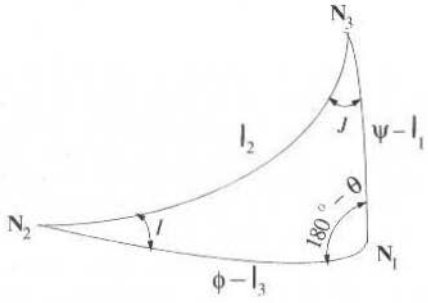


Fig. 2 Spherical Triangle $N_1N_2N_3$

with Andoyer's variables and Euler angles

The problem of optimization is formulated, with the dynamic system describing the rotational motion of the satellite given by eqs. (1), considering the torques provided by a power limited propulsion system and the gravity gradient torque.

Optimization of the Attitude's Control

The optimization problem of attitude control is initially introduced by Lagrange formulation, with the dynamical system describing the rotational motion of the satellite. The torques provided by a power limited propulsion system and the gravity gradient torque are included. The state is defined by Andoyer's variables ℓ_i , L_i , $i = 1,2,3$, and the control by the components of non-conservative external torques in the artificial satellite's principal axes of inertia (Oxyz). The Mayer problem, without constraints on control variables and fixed initial time t_0 and fixed final time t_f, is also analyzed here.

The state equations are the equations of the rotational motion of the satellite, including external torques, in the extended canonical form. For a satellite with cylindrical symmetry, they are given by :

$$\frac{d\ell_1}{dt} = \left(\frac{1}{C}-\frac{1}{A}\right)L_1 + f_1(\ell_i, L_i) + C_{1x}Q_x + C_{1y}Q_y$$

$$\frac{d\ell_2}{dt} = \frac{1}{A}L_2 + f_2(\ell_i, L_i) + C_{2x}Q_x + C_{2y}Q_y + C_{2z}Q_z$$

$$\frac{d\ell_3}{dt} = f_3(\ell_i, L_i) + C_{3x}Q_x + C_{3y}Q_y + C_{3z}Q_z \qquad (3)$$

$$\frac{dL_1}{dt} = Q_z$$

$$\frac{dL_2}{dt} = f_5(\ell_i, L_i) + C_{5x}Q_x + C_{5y}Q_y + C_{5z}Q_z$$

$$\frac{dL_3}{dt} = f_6(\ell_i, L_i) + C_{6x}Q_x + C_{6y}Q_y + C_{6z}Q_z ,$$

where Q_x, Q_y *and* Q_z are the components of propulsive torque **Q** on the system Oxyz and $f_j(\ell_i, L_i)$, j = 1,...,6, i=1,2,3, are functions related with the gravity gradient torque and C_{jx}, C_{jy} and C_{jz}, j = 1,...,6, can be found in Zanardi (Zanardi, et al., 1994).

The performance index J, associated with the fuel consumption, is introduced by:

$$J = \frac{1}{2}\int Q^2 dt \quad , \qquad (4)$$

where Q is the magnitude of propulsive torque Q.

The optimization problem consists in determining the optimal control Q^*, which transfers the space vehicle from the initial state (ℓ_{i0}, L_{i0}) at t_0 to the final state (ℓ_{if}, L_{if}) at t_f, such that the consumption is a minimum.

The Mayer problem associated to the minimization of fuel consumption during the attitude maneuvers will be defined as follows. The state vector **x** is defined by the Andoyer's variables and the performance index J,

$$x = [\ell_1 \ \ell_2 \ \ell_3 \ L_1 \ L_2 \ L_3 \ J]^T . \qquad (5)$$

The dynamic system is described by equations (3) and by the differential equation associated with J:

$$\frac{dJ}{dt} = \frac{1}{2}\left[Q_x^2 + Q_y^2 + Q_z^2\right]. \qquad (6)$$

This system can be represent by:

$$\frac{dx}{dt} = F(x, Q). \qquad (7)$$

The performance index is now given by:

$$J_f = J(t_f) , \tag{8}$$

and the boundary conditions are determined by the initial state $\mathbf{x}_0 = (\ell_{i0}, L_{i0}, 0)$ and the final state

$$x_f = (\ell_{if}, L_{if}, J_f) , \quad i=1,2,3 .$$

Following the Pontryagin Maximum Principle (Pontryagin, et al., 1962), the adjoint vector $\boldsymbol{p}_{\mathbf{x}}$ is introduced and the Hamiltonian function is formed using eqs. (3) and (6):

$$\begin{aligned} H(x, \boldsymbol{p}_{\mathbf{x}}, Q) = & \left[\frac{1}{C} - \frac{1}{A}\right] L_1 p_1 + \frac{1}{A} L_2 p_2 + \\ & + p_1 f_1(\ell_i, L_i) + p_2 f_2(\ell_i, L_i) + p_3 f_3(\ell_i, L_i) + \\ & + P_2 f_5(\ell_i, L_i) + P_3 f_6(\ell_i, L_i) + p_1 [C_{1x} Q_x + \\ & + C_{1y} Q_y] + p_2 [C_{2x} Q_x + C_{2y} Q_y + C_{2z} Q_z] + \\ & + p_3 [C_{2x} Q_x + C_{2y} Q_y + C_{3z} Q_z] + P_1 Q_z + \\ & + P_2 [C_{5x} Q_x + C_{5y} Q_y + C_{5z} Q_z] + P_3 [C_{6x} Q_x + \\ & + C_{6y} Q_y + C_{6z} Q_z] + \frac{1}{2} P_J [Q_x^2 + Q_y^2 + Q_z^2] . \end{aligned} \tag{9}$$

The optimal torques Q* must be selected so that the Hamiltonian function reaches its maximum value:

$$Q^* = arg \ max \, H(x, p_x, Q) .$$

The equations of control (stationary conditions) are given by:

$$\begin{aligned} & p_1 C_{1x} + p_2 C_{2x} + p_3 C_{3x} + P_2 C_{5x} + P_J Q_x = 0 \\ & p_1 C_{1y} + p_2 C_{2y} + p_3 C_{3y} + P_2 C_{5y} + P_3 C_{6y} + P_J Q_y = 0 \\ & p_2 C_{2z} + p_3 C_{3z} + P_1 + P_2 C_{5z} + P_3 C_{6z} + P_J Q_z = 0 . \end{aligned} \tag{10}$$

Solving the system (10), the control Q*, is given by:

$$\begin{aligned} Q_x^* &= -\frac{p_1 C_{1x} + p_2 C_{2x} + p_3 C_{3x} + P_2 C_{5x} + P_3 C_{6x}}{P_J} \\ Q_y^* &= -\frac{p_1 C_{1y} + p_2 C_{2y} + p_3 C_{3y} + P_2 C_{5y} + P_3 C_{6y}}{P_J} \\ Q_z^* &= -\frac{p_2 C_{2z} + p_3 C_{3z} + P_1 + P_2 C_{5z} + P_3 C_{6z}}{P_J} , \end{aligned} \tag{11}$$

and the maximum Hamiltonian function H^*, computed by (9) and (10), can be expressed by:

$$H^* = H^*(x, p_x). \tag{12}$$

The adjoint variable P_J is a first integral of the canonical system defined by H^* and its value, obtained from the transversality conditions, is equal -1. Consequently, the order of the dynamic system which describes the problem is reduced and the Hamiltonian assumes the form:

$$H^* = H_0^* + H_G^* + H_Q^*, \tag{13}$$

where:

$$H_0^* = \left[\frac{1}{C} - \frac{1}{A}\right] L_1 p_1 + \frac{1}{A} L_2 p_2 , \tag{14}$$

$$H_G^* = \sum_{i=1}^{3} \left[p_i f_i + P_i f_{i+3} \right], \tag{15}$$

$$\begin{aligned} H_Q^* = {} & \frac{p_1^2}{2 L_2^2 \sin^2 J} + \frac{p_2^2}{2L_2^2}\left[\cot^2 I + \cot^2 J + \frac{1}{2}P_2^2 + \frac{1}{2}P_3^2 - \frac{p_1 p_2}{L_2^2 \sin J}\left[\cot J + \cot I \cos \ell_2\right] + \right. \\ & + \frac{p_1 p_3 \cos \ell_2}{L_2^2 \sin I \sin J} + \frac{p_1 P_3}{2L_2 \sin J} \sum_{\varepsilon} \varepsilon \cos(I + \varepsilon \ell_2) - \frac{p_2 p_3}{L_2^2 \sin I}\left[\cot I + \cot J \cos \ell_2\right] + \\ & + \frac{p_2 P_1}{2L_2} \cot I \sum_{\varepsilon} \varepsilon \cos(J + \varepsilon \ell_2) + \frac{p_2 P_3}{2L_2}\left\{ \cot J \sum_{\varepsilon} \varepsilon \cos(I + \varepsilon \ell_2) - \right. \\ & \left. - \frac{1}{2}\cot I \left[\sum_{\varepsilon} \varepsilon \cos(I + 2\varepsilon \ell_2) + \sum_{\varepsilon,\delta} \varepsilon\delta \cos(I + 2\delta + \varepsilon \ell_2) - \frac{1}{2} \sum_{\varepsilon,\delta} \delta \cos(I + 2\varepsilon J + 2\delta \ell_2) \right] \right\} + \\ & + \frac{p_3 P_1}{2L_2 \sin I} \sum_{\varepsilon} \varepsilon \cos(J + \varepsilon \ell_2) + \frac{p_3 P_3}{L_2^2}\left[\frac{1}{2}\sin 2\ell_2 + \frac{1}{4} \sum_{\varepsilon} \varepsilon \sin 2(J + \varepsilon \ell_2) + \frac{1}{\sin I} \sum_{\varepsilon,\delta} \varepsilon\delta \cos(I + 2\delta J + \right. \\ & \left. + \varepsilon \ell_2) \right] + P_1 P_2 \cos J - \frac{1}{2} P_1 P_3 \left[\cos(I + J) + \cos(I - J) + \frac{1}{2} \sum_{\varepsilon,\delta} \cos(I + \delta J + \varepsilon \ell_2) \right] - \\ & \left. - \frac{1}{2} P_2 P_3 \left[\cos(I + 2J) + \cos(I - 2J) + \frac{1}{2} \sum_{\varepsilon,\delta} \varepsilon \cos(I + 2\varepsilon J + \delta \ell_2) \right] \right], \end{aligned} \tag{16}$$

where $\underset{\varepsilon}{\smile}$ and $\underset{\varepsilon,\delta}{\smile}$ means that ε and δ assume the values: +1 and -1; H_0^* is the unperturbed Hamiltonian associated with the torque-free rotational motion, H_G^* and H_Q^* are the perturbed functions. H_G^* is associated with the gravity gradient torque, and H_Q^* is related with optimal control. The two first parcels of the eq. (13) is associated with the conservative system : $(H_0^* + H_G^*)$ and the other parcel (H_Q^*) is associated with the non-conservative system.

Attitude Optimal Corrections

The problem of optimal attitude corrections, with fixed duration, given by the linearized Hamiltonian around the reference attitude is considered.

The reference attitude is given by the solution of the torque-free rotational motion (Zanardi, 1986) (Deprit, 1970) :

$$\bar{\ell}_1 = \bar{\ell}_{10} + \bar{\omega}_1 (t - t_0)$$

$$\bar{\ell}_2 = \bar{\ell}_{20} + \bar{\omega}_2 (t - t_0) \qquad (17)$$

$$\bar{\ell}_3 = \bar{\ell}_{30} \qquad \bar{L}_i = \bar{L}_{i0}$$

$i = 1,2,3$;

where the bar means reference attitude and, $\bar{L}_1 = \bar{L}_2 \cos \bar{J}$, $\bar{L}_3 = \bar{L}_2 \cos \bar{I}$, $\bar{\omega}_1 = \left[\frac{1}{C} - \frac{1}{A}\right]\bar{L}_1$ and $\bar{\omega}_2 = \frac{1}{A}\bar{L}_2$.

The analytical solution for the attitude maneuvers, considering only the propulsive torque, is described by following Hamiltonian:

$$H^* = H_0^* + H_Q^* \ , \qquad (18)$$

with H_0^* and H_Q^* defined by eqs (13) and (15), respectively, and linearized around the reference attitude defined by (17).

A first order analytical solution for the problem of attitude corrections can be gotten by simple quadratures. This solution will be expressed by:

$$\Delta x = E \ p_{x_0} + D \ , \qquad (19)$$

where : Δx represents the variations determined over the Andoyer's variables; p_{x_0} means the initial values of the adjoint variables; E is a 6x6 matrix, related with the propulsive system and D is a 6x1 vector, related with the solution of non-perturbed problem. The adjoints variables are first integral of the canonical system, associated by the maximum Hamiltonian H^-, linearized around the reference attitude.

The solution (19) represents a complete solution for the attitude optimal corrections and contains secular terms and short periodic terms. The six integration constants must be determined in order to satisfy the two-point boundary value problem, which consists in starting in initial condition A_0 at initial time and reaching the final attitude A_1 at fixed final time .

The non-null elements of the matrices D and E, are given by:

$$d_1 = \bar{\omega}_1 \, \Delta t \qquad d_2 = \bar{\omega}_2 \, \Delta t$$

$$e_{11} = \frac{\Delta t}{\bar{L}_{20}^2 \sin^2 \bar{J}} \qquad e_{14} = \frac{1}{2}\left[\frac{1}{C} - \frac{1}{A}\right]\Delta t^2$$

$$e_{15} = \frac{1}{2}\left[\frac{1}{C} - \frac{1}{A}\right]\cos \bar{J} \, \Delta t^2 \qquad e_{25} = \frac{1}{2A}\Delta t^2$$

$$e_{31} = \frac{\sin \bar{\ell}_2 - \sin \bar{\ell}_{20}}{\bar{L}_{20}^2 \, \bar{\omega}_2 \, \sin \bar{I} \, \sin \bar{J}} \qquad e_{33} = \frac{\Delta t}{\bar{L}_{20}^2 \sin^2 \bar{I}}$$

$$e_{44} = e_{55} = e_{66} = \Delta t \qquad e_{45} = e_{54} = \cos \bar{J} \, \Delta t$$

$$e_{12} = \frac{1}{\bar{L}_{20}^2}\left[\frac{\cos\bar{J}\Delta t}{\sin^2\bar{J}} + \frac{\cot\bar{I}}{\bar{\omega}_2\sin\bar{J}}\left(\sin\bar{\ell}_2 - \sin\bar{\ell}_{20}\right)\right] - \frac{\cot\bar{I}}{2\bar{L}_{20}\bar{\omega}_2^2}\left[\frac{1}{C}-\frac{1}{A}\right]\sum_{\varepsilon}\left[\cos(\bar{J}+\varepsilon\,\bar{\ell}_2) - \cos(\bar{J}+\varepsilon\,\bar{\ell}_{20})\right]$$

$$e_{13} = -\frac{\cot\bar{I}}{2\bar{L}_{20}\bar{\omega}_2^2\sin\bar{I}}\left[\frac{1}{C}-\frac{1}{A}\right]\sum_{\varepsilon}\left[\cos(\bar{J}+\varepsilon\,\bar{\ell}_2) - \cos(\bar{J}+\varepsilon\,\bar{\ell}_{20})\right] - \frac{\sin\bar{\ell}_2 - \sin\bar{\ell}_{20}}{\bar{L}_{20}^2\bar{\omega}_2\sin\bar{I}\,\sin\bar{J}}$$

$$e_{16} = -\frac{1}{4}\left[\frac{1}{C}-\frac{1}{A}\right]\left\{\left[\cos(\bar{I}+\bar{J}) + \cos(\bar{I}-\bar{J})\right]\Delta t^2 - \frac{1}{\bar{\omega}_2^2}\sum_{\varepsilon\,\delta}\left[\cos(\bar{I}+\delta\,\bar{J}+\varepsilon\,\bar{\ell}_2) - \cos(\bar{I}+\delta\,\bar{J}+\varepsilon\,\bar{\ell}_{20})\right]\right\} +$$

$$+ \frac{1}{2\bar{L}_{20}\bar{\omega}_2\sin\bar{I}}\sum_{\varepsilon}\left[\sin(\bar{I}+\varepsilon\,\bar{\ell}_2) - \sin(\bar{I}+\varepsilon\,\bar{\ell}_{20})\right]$$

$$e_{21} = -\frac{1}{\bar{L}_{20}^2\sin\bar{J}}\left[\frac{\cos\bar{J}\Delta t}{\sin\bar{J}} + \frac{\cot\bar{I}}{\bar{\omega}_2}\left(\sin\bar{\ell}_2 - \sin\bar{\ell}_{20}\right)\right]$$

$$e_{22} = \frac{2}{\bar{L}_{20}^2\bar{\omega}_2}\cot\bar{I}\,\cot\bar{J}\left(\operatorname{sen}\bar{\ell}_2 - \operatorname{sen}\ell_{20}\right) + \frac{1}{\bar{L}_{20}^2}\left[\cot^2\bar{I} + \cot^2\bar{J}\right]\Delta t$$

$$e_{23} = e_{32} = -\frac{1}{\bar{L}_{20}^2\sin\bar{J}}\cot\bar{I}\,\Delta t + \frac{1}{\bar{L}_{20}^2\bar{\omega}_2\sin\bar{J}}\cot\bar{J}\left(\sin\bar{\ell}_2 - \sin\ell_{20}\right)$$

$$e_{24} = \frac{\cot\bar{I}}{2\bar{L}_{20}\bar{\omega}_2}\sum_{\varepsilon}\left[\sin(\bar{J}+\varepsilon\,\bar{\ell}_2) - \sin(\bar{J}+\varepsilon\,\bar{\ell}_{20})\right] + \frac{1}{2A}\cos\bar{J}\,\Delta t^2$$

$$e_{26} = -\frac{1}{4A}\left[\frac{1}{\bar{\omega}_2^2}\sum_{\varepsilon\,\delta}\left[\sin(\bar{I}+2\varepsilon\,\bar{J}+\delta\,\bar{\ell}_2) - \sin(\bar{I}+2\varepsilon\,\bar{J}+\delta\,\bar{\ell}_{20})\right] - \left[\cos(\bar{I}+2\bar{J}) + \cos(\bar{I}-2\bar{J})\right]\Delta t^2\right] +$$

$$+ \frac{\cot\bar{J}}{2\bar{L}_{20}\bar{\omega}_2}\sum_{\varepsilon}\left[\sin(\bar{I}+\varepsilon\,\bar{\ell}_2) - \sin(\bar{I}+\varepsilon\,\bar{\ell}_{20})\right] - \frac{\cot\bar{I}}{4\bar{L}_{20}\bar{\omega}_2}\left[\frac{1}{2}\sum_{\varepsilon}\left[\sin(\bar{I}+2\varepsilon\,\bar{\ell}_2) - \sin(\bar{I}+2\varepsilon\,\bar{\ell}_{20})\right] +\right.$$

$$\left. + \sum_{\varepsilon\,\delta}\left[\sin(\bar{I}-2\delta\,\bar{J}+\varepsilon\,\bar{\ell}_2) - \sin(\bar{I}-2\delta\,\bar{J}+\varepsilon\,\bar{\ell}_{20})\right] + \frac{1}{4}\sum_{\varepsilon\,\delta}\left[\sin(\bar{I}+2\varepsilon\,\bar{J}+2\delta\,\bar{\ell}_2) - \sin(\bar{I}+2\varepsilon\,\bar{J}+2\delta\,\bar{\ell}_{20})\right]\right.$$

$$e_{34} = e_{43} = \sum_{\varepsilon}\frac{\left[\sin(\bar{J}+\varepsilon\,\bar{\ell}_2) - \sin(\bar{J}+\varepsilon\,\bar{\ell}_{20})\right]}{2\bar{L}_{20}\bar{\omega}_2\sin\bar{I}}$$

$$e_{36} = e_{63} = \frac{1}{\bar{L}_{20}\bar{\omega}_2}\left[-\frac{1}{8}\sum_{\varepsilon}\left[\cos2(\bar{J}+\varepsilon\,\bar{\ell}_2) - \cos2(\bar{J}+\varepsilon\,\bar{\ell}_{20})\right] - \frac{1}{\sin\bar{I}}\sum_{\varepsilon\,\delta}\left[\sin(\bar{I}+2\delta\,\bar{J}+\varepsilon\,\bar{\ell}_2) -\right.\right.$$

$$\left.\left. - \sin(\bar{I}+2\delta\,\bar{J}+\varepsilon\,\bar{\ell}_{20}) - \frac{1}{4}\left(\cos2\bar{\ell}_2 - \cos2\bar{\ell}_{20}\right)\right.\right.$$

$$e_{42} = \frac{\cot \bar{I}}{\bar{L}_{20}\,\bar{\omega}_2} \sum_{\varepsilon} \left[\sin(\bar{J} + \varepsilon\,\bar{\ell}_2) - \sin(\bar{J} + \varepsilon\,\bar{\ell}_{20}) \right]$$

$$e_{46} = e_{64} = -\frac{1}{2}\left[\cos(\bar{I} + \bar{J}) + \cos(\bar{I} - \bar{J}) \right] \Delta t + \frac{1}{2\bar{\omega}_2} \sum_{\varepsilon}\sum_{\delta} \left[\sin(\bar{I} + \delta\,\bar{J} + \varepsilon\,\bar{\ell}_2) - \sin(\bar{I} + \delta\,\bar{J} + \varepsilon\,\bar{\ell}_{20}) \right]$$

$$e_{56} = e_{65} = -\frac{1}{2}\left[\cos(\bar{I} + 2\bar{J}) + \cos(\bar{I} - 2\bar{J}) \right] \Delta t + \frac{1}{4\bar{\omega}_2} \sum_{\varepsilon}\sum_{\delta} \left[\sin(\bar{I} + \delta\,\bar{J} + \varepsilon\,\bar{\ell}_2) - \sin(\bar{I} + \delta\,\bar{J} + \varepsilon\,\bar{\ell}_{20}) \right]$$

$$e_{61} = \frac{1}{2\bar{L}_{20}\bar{\omega}_2 \sin \bar{J}} \sum_{\varepsilon} \left[\sin(\bar{I} + \varepsilon\,\bar{\ell}_2) - \sin(\bar{I} + \varepsilon\,\bar{\ell}_{20}) \right]$$

$$e_{62} = \frac{\cot \bar{J}}{2\bar{L}_{20}\bar{\omega}_2} \sum_{\varepsilon} \left[\sin(\bar{I} + \varepsilon\,\bar{\ell}_2) - \sin(\bar{I} + \varepsilon\,\bar{\ell}_{20}) \right] - \frac{\cot \bar{I}}{4\bar{L}_{20}\bar{\omega}_2} \left[\frac{1}{2} \sum_{\varepsilon} \left[\sin(\bar{I} + 2\varepsilon\,\bar{\ell}_2) - \sin(\bar{I} + 2\varepsilon\,\bar{\ell}_{20}) \right] - \right.$$

$$\left. - \frac{1}{4} \sum_{\varepsilon}\sum_{\delta} \left[\sin(\bar{I} - 2\varepsilon\,\bar{J} + \delta\,\bar{\ell}_2) - \sin(\bar{I} - 2\varepsilon\,\bar{J} + \delta\,\bar{\ell}_{20}) \right] \right] - \sum_{\varepsilon}\sum_{\delta} \left[\sin(\bar{I} - 2\delta\,\bar{J} + \varepsilon\,\bar{\ell}_2) - \sin(\bar{I} - 2\delta\,\bar{J} + \varepsilon\,\bar{\ell}_{20}) \right], \tag{20}$$

where Δt is the duration of the maneuver.

For artificial satellite with spherical symmetry, A = C, the analytical solutions given by (18) to (20) simplifies, since H_0^* is computed by:

$$H_0^* = \frac{\bar{L}_{20}}{A} p_2, \tag{21}$$

and $d_1 = 0$, $e_{12} = e_{21}$, $e_{13} = e_{31}$, $e_{16} = e_{61}$, $e_{14} = e_{15} = 0$.

The attitude's corrections maneuvers problem can be simplified if we consider only the secular terms. The Hamiltonian function associated to long duration maneuvers is described by Eqs. (14) and (18), with the perturbed Hamiltonian given by:

$$H_Q^* = \frac{1}{2\bar{L}_{20}^2} \left\{ \frac{p_{10}^2}{\sin^2 \bar{J}} + p_{20}^2 \left[\cot^2 \bar{I} + \cot^2 \bar{J} \right] + \frac{p_{30}^2}{\sin^2 \bar{I}} \right\} + \frac{1}{2}\left[P_{10}^2 + P_{20}^2 + P_{30}^2 \right] - \frac{p_{20}}{\bar{L}_{20}^2} \left[\frac{p_{10} \cot \bar{J}}{\sin \bar{J}} + \right.$$

$$\left. + \frac{p_{30} \cot \bar{I}}{\sin \bar{I}} \right] + P_{10} P_{20} \cos \bar{J} - \frac{1}{2} P_{30} \left\{ P_{10} \left[\cos(\bar{I} + \bar{J}) + \cos(\bar{I} - \bar{J}) \right] - P_{20} \left[\cos(\bar{I} + 2\bar{J}) + \cos(\bar{I} - 2\bar{J}) \right] \right\}. \tag{22}$$

The differential equations for the maneuvers are:

$$\frac{d\ell_1}{dt} = \left[\frac{1}{C} - \frac{1}{A}\right]\left[\bar{L}_{10} + \frac{d}{dt}\left(t\Delta L_1\right) - t\frac{dL_1}{dt}\right] + \frac{p_{10}}{\bar{L}_{20}^2 \sin^2 \bar{J}} - \frac{p_{20}\cot \bar{J}}{\bar{L}_{20}^2 \sin \bar{J}}$$

$$\frac{d\ell_2}{dt} = \frac{1}{A}\left[\bar{L}_{20} + \frac{d}{dt}\left(t\Delta \bar{L}_2\right) - t\frac{dL_2}{dt}\right] - \frac{p_{10}\cot \bar{J}}{\bar{L}_{20}^2 \sin \bar{J}} + \frac{p_{20}}{\bar{L}_{20}^2}\left[\cot^2 \bar{I} + \cot^2 \bar{J}\right] - \frac{p_{30}\cot \bar{I}}{\bar{L}_{20}^2 \sin \bar{I}}$$

$$\frac{d\ell_3}{dt} = -\frac{p_{20}\cot \bar{I}}{\bar{L}_{20}^2 \sin \bar{I}} + \frac{p_{30}}{\bar{L}_{20}^2 \sin^2 \bar{I}}$$

$$\frac{dL_1}{dt} = P_{10} + P_{20}\cos \bar{J} - \frac{1}{2}P_{30}\left[\cos(\bar{I} + \bar{J}) + \cos(\bar{I} - \bar{J})\right]$$

$$\frac{dL_2}{dt} = P_{10}\cos \bar{J} + P_{20} - \frac{1}{2}P_{30}\left[\cos(\bar{I} + 2\bar{J}) + \cos(\bar{I} - 2\bar{J})\right]$$

$$\frac{dL_3}{dt} = -\frac{1}{2}P_{10}\left[\cos(\bar{I} + \bar{J}) + \cos(\bar{I} - \bar{J})\right] - \frac{1}{2}P_{20}\left[\cos(\bar{I} + 2\bar{J}) + \cos(\bar{I} - 2\bar{J})\right] + P_{30}. \quad (23)$$

The simplified analytical solutions for eqs. (23) are given in the matrix form as:

$$\Delta x = E_S\, p_{x0} + D_S, \quad (24)$$

where the subscript S means the secular part, with non-null elements of the matrices D_S and E_S are presented in Santos (Santos, et al., 1999).

The analytical solutions for the optimal corrections of the artificial satellite with spherical symmetry, whose Hamiltonian is described in the eqs (18), (21) and (22), are also given by the matrix form (24), but $D_S = [0 \;\; d_2 \;\; 0 \;\; 0 \;\; 0 \;\; 0]^t$, with $d_2 = \omega_2 \Delta t$ and $e_{14} = e_{15} = e_{16} = 0$.

If we consider the satellite with the cylindrical symmetry and the influence of the gravity gradient torque, the Hamiltonian function will be given by (13), (14) e (22). The Hamiltonian function H_G^* associated with the gravity gradient torque, taking in account terms up to the inverse of the cubic of the distance between the Earth's center of mass and the satellite's center of mass and assuming circular orbit, the orbital inclination and the longitude of ascending node equal zero degree, is given by:

$$H_G^* = \frac{3}{4\bar{L}_{20}}\frac{\mu}{r^3}[C - A]\left\{ p_{10}\left[1 - 3\cos^2 \bar{I}\right]\cos \bar{J} - \right.$$

$$\left. - p_{20}\left[\left[1 - 3\cos^2 \bar{I}\right]\cos^2 \bar{J} + \left[1 - 3\cos^2 \bar{J}\right]\cos^2 \bar{I}\right] + p_{30}\left[1 - 3\cos^2 \bar{J}\right]\cos \bar{I} \right\}, \quad (25)$$

where μ is the Gaussian constant and r is the radial distance.

If the satellite's orbit is circular and the gravity gradient torque is taken in account during the long duration maneuvers, the analytical solutions are also given by eq. (24), but the elements of $D_S = [d_1 \;\; d_2 \;\; d_3 \;\; 0 \;\; 0 \;\; 0]^t$ are expressed by:

$$d_1 = \left[\frac{1}{C} - \frac{1}{A}\right]\bar{L}_{10} + \frac{3\mu}{4r^3}\left(\frac{C-A}{\bar{L}_{20}}\right)\left[1 - 3\cos^2\bar{I}\right]\cos\bar{J}\Delta t$$

$$d_2 = \frac{1}{A}\bar{L}_{20} - \frac{3\mu}{4r^3}\left(\frac{C-A}{\bar{L}_{20}}\right)\left[1 - 3\cos^2\bar{I}\right]\cos^2\bar{J} + \left[1 - 3\cos^2\bar{J}\right]\cos^2\bar{I}\Delta t \qquad (26)$$

$$d_3 = \frac{3\mu}{4r^3}\left(\frac{C-A}{\bar{L}_{20}}\right)\left[1 - 3\cos^2\bar{J}\right]\cos\bar{I}\Delta t\,.$$

For elliptic orbit, the Hamiltonian H_G^*, taking in account terms up to second order in eccentricity and assuming the orbital inclination and the longitude of ascending node equal zero degree, has the following form:

$$H_G^* = \frac{3\mu}{a^3}\left(\frac{C-A}{4\bar{L}_{20}}\right)\left\{1 + \frac{3}{2}e^2\right\}\left\{p_{10}\left[1 - 3\cos^2\bar{I}\right]\cos\bar{J}\right.$$

$$\left. - p_{20}\left[\left[1 - 3\cos^2\bar{I}\right]\cos^2\bar{J} + \left[1 - 3\cos^2\bar{J}\right]\cos^2\bar{I}\right] + p_{30}\left[1 - 3\cos^2\bar{J}\right]\cos\bar{I}\right\}. \qquad (27)$$

where a is the semi-major axis and e is the eccentricity. In this case, the analytical solution has also the matrix form (24), with $D_S = [d_1 \;\; d_2 \;\; d_3 \;\; 0 \;\; 0 \;\; 0]'$ and

$$d_1 = \frac{3\mu}{4a^3}\left(\frac{C-A}{\bar{L}_{20}}\right)\left\{1 + \frac{3}{2}e^2\right\}\left[1 - 3\cos^2\bar{I}\right]\cos\bar{J}\Delta t + \left[\frac{1}{C} - \frac{1}{A}\right]\bar{L}_{10}$$

$$d_2 = \frac{3\mu}{4a^3}\left(\frac{C-A}{\bar{L}_{20}}\right)\left\{1 + \frac{3}{2}e^2\right\}\left[\left[1 - 3\cos^2\bar{I}\right]\cos^2\bar{J} + \right. \qquad (28)$$

$$\left. + \left[1 - 3\cos^2\bar{J}\right]\cos^2\bar{I}\right]\Delta t + \frac{1}{A}\bar{L}_{20}$$

$$d_3 = \frac{3\mu}{4a^3}\left(\frac{C-A}{\bar{L}_{20}}\right)\left\{1 + \frac{3}{2}e^2\right\}\left[1 - 3\cos^2\bar{J}\right]\cos\bar{I}\Delta t\,.$$

The Optimal Consumption

The optimal consumption J that is necessary for the satellite's attitude correction is obtained by quadrature of the equation:

$$\dot{J} = \frac{1}{2}\,Q^{*2}\,, \qquad (29)$$

where the components of the optimal torques are expressed in terms of the Andoyer's variables and their adjoints by:

$$Q_x^* = \frac{p_1 \cos \ell_1}{\bar{L}_2 \sin \bar{J}} - \frac{p_2}{\bar{L}_2}\left[\cos\ell_1 \cot \bar{J} + \cot \bar{I}(\cos\ell_1 \cos\ell_2 - \sin\ell_1 \sin\ell_2 \cos\bar{J})\right] - \frac{p_3}{\bar{L}_2 \sin \bar{I}}(\cos\ell_1 \cos\ell_2 -$$

$$- \sin\ell_1 \sin\ell_2 \cos\bar{J}) + P_2 \sin\bar{J} \sin\ell_1 + P_3\left[\sin\bar{I}\cos\ell_1 \sin\ell_2 + \sin\ell_1(\cos\bar{I}\sin\bar{J} + \sin\bar{I}\cos\bar{J}\cos\ell_2)\right]$$

$$Q_y^* = \frac{p_1 \sin \ell_1}{\bar{L}_2 \sin \bar{J}} + \frac{p_2}{\bar{L}_2}\left[\sin\ell_1 \cot \bar{J} + \cot \bar{I}(\sin\ell_1 \cos\ell_2 + \cos\ell_1 \sin\ell_2 \cos\bar{J})\right] - \frac{p_3}{\bar{L}_2 \sin \bar{I}}(\sin\ell_1 \cos\ell_2 +$$

$$+ \cos\ell_1 \sin\ell_2 \cos\bar{J}) + P_2 \sin\bar{J} \cos\ell_1 + P_3\left[-\sin\bar{I} \sin\ell_1 \sin\ell_2 + \cos\ell_1(\cos\bar{I} \sin\bar{J} + \sin\bar{I}\cos\bar{J}\cos\ell_2)\right]$$

$$Q_z^* = \frac{p_2}{\bar{L}_2} \sin\ell_2 \sin\bar{J} \cot \bar{I} + \frac{p_3}{\bar{L}_2 \sin \bar{I}} \sin\ell_2 \text{sen}\bar{J} + P_1 + P_2 \cos\bar{J} + P_3(\cos\ell_2 \sin\bar{I} \sin\bar{J} - \cos\bar{I} \cos\bar{J}) . \tag{30}$$

In terms of the elements of the matrix E and the initial values of the adjoint variables, the optimal consumption is given by:

$$\Delta J = \frac{1}{2}e_{11}p_{10}^2 + \frac{1}{2}e_{22}p_{20}^2 + \frac{1}{2}e_{33}p_{30}^2 + \frac{1}{2}e_{44}P_{10}^2 + \frac{1}{2}e_{55}P_{20}^2 + \frac{1}{2}e_{66}P_{30}^2 + e_{21}\,p_{10}p_{20} + e_{31}\,p_{10}p_{30} +$$

$$+ e_{61}p_{10}P_{30} + e_{23}p_{20}p_{30} + e_{34}p_{30}P_{10} + e_{36}p_{30}P_{30} + e_{42}p_{20}P_{10} + e_{45}P_{10}P_{20} + e_{46}P_{10}P_{30} +$$

$$+ e_{56}P_{20}P_{30} + e_{62}p_{20}P_{30} . \tag{31}$$

In the case where only the secular terms are considered, the eq. (31) is simplified and given in terms of the elements of the matrix E_S:

$$\Delta J_S = \frac{1}{2}e_{11}p_{10}^2 + \frac{1}{2}e_{22}p_{20}^2 + \frac{1}{2}e_{33}p_{30}^2 + \frac{1}{2}e_{44}P_{10}^2 + \frac{1}{2}e_{55}P_{20}^2 + \frac{1}{2}e_{66}P_{30}^2 + e_{12}\,p_{10}p_{20} + e_{23}\,p_{20}p_{30} +$$

$$+ e_{45}P_{10}P_{20} + e_{46}P_{10}P_{30} + e_{56}P_{20}P_{30} . \tag{32}$$

Numerical Simulation

In this section, numerical results are presented for a long-time attitude maneuver of a cylindrical satellite. The effects of gravity gradient torque are not included in this simulation. The minimal consumption, the magnitudes of the optimal torques and the evolution of the analytical solution are computed by using Eqs. (32), (11) and (24), respectively. The physical characteristic of the satellite, the initial state and the final state are presented in the following tables. The initial values for the adjoint variables are gotten numerically solving the algebraic system (24).

The temporal evolutions of the Euler angles (φ , θ ,ψ) and their rate variations ($\dot{\varphi}$,$\dot{\theta}$,$\dot{\psi}$) are shown in the Figs. 3 and 4, respectively, during the attitude maneuver . The propagations of the Andoyer variables ($\ell_1, \ell_2, \ell_3, L_1, L_2, L_3$)are shown in the Figs. 5 and 6. The magnitudes of the optimal torques and the magnitude of the performance index are presented in the Figs. 7 and 8, respectively.

Table 1 Physical characteristic

Initial mass = 1.0×10^3 kg
Principal Moments of inertia
A = B = 3.95×10^2 kg m^2
C = 1.05×10^2 kg m^2

Table 2 Initial and Final values for the Euler Angles and their Rate Variations

Initial Attitude t_0 = 0.0 s	Final Attitude t_f = 5.0×10^4 s
$\varphi_0 = 1.0 \times 10^1$ deg	$\varphi_f = 1.3 \times 10^1$ deg
$\theta_0 = 1.5 \times 10^1$ deg	$\theta_f = 1.7 \times 10^1$ deg
$\psi_0 = 1.0 \times 10^1$ deg	$\psi_f = 1.1 \times 10^1$ deg
$\dot{\varphi}_0 = 2.0 \times 10^{-3}$ deg /s	$\dot{\varphi}_f = 1.7 \times 10^{-3}$ deg /s
$\dot{\theta}_0 = 2.0 \times 10^{-3}$ deg /s	$\dot{\theta}_f = 2.0 \times 10^{-3}$ deg /s
$\dot{\psi}_0 = .2.0 \times 10^{-3}$ deg /s	$\dot{\psi}_f = 1.5 \times 10^{-3}$ deg /s

Table 3 Initial Adjoint Variables

$p_{10} = -3.858 \times 10^{-1}$
$p_{20} = -6.671 \times 10^{-1}$
$p_{30} = 1.0 \times 10^{-2}$
$P_{10} = -1.91 \times 10^{-2}$
$P_{20} = -1.91 \times 10^{-2}$
$P_{30} = -1.78 \times 10^{-2}$
$P_{J0} = -1.0$

Table 4 Terminal Values for the Andoyer's Variables

Initial Attitude t_0 = 0 s	Final Attitude t_f= 5×10^4 s
$\ell_{10} = 8.55 \times 10^1$ deg	$\ell_{1f} = 8.70 \times 10^1$ deg
$\ell_{20} = 1.97 \times 10^2$ deg	$\ell_{2f} = 1.98 \times 10^2$ deg
$\ell_{30} = 1.07 \times 10^2$ deg	$\ell_{3f} = 1.1 \times 10^2$ deg
$L_{10} = 2.35 \times 10^{-1}$ kg m^2 / s	$L_{1f} = 1.87 \times 10^{-1}$ kg m^2 / s
$L_{20} = 5.21 \times 10^{-1}$ kg m^2 / s	$L_{2f} = 5.00 \times 10^{-1}$ kg m^2 / s
$L_{30} = 2.57 \times 10^{-1}$ kg m^2 / s	$L_{3f} = 2.12 \times 10^{-1}$ kg m^2 / s

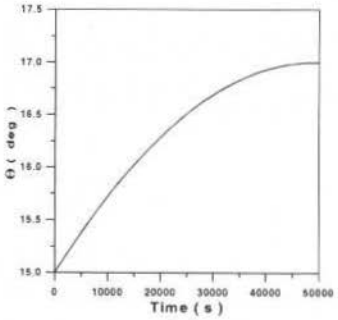


Fig. 3a Evolution of Euler Angle θ

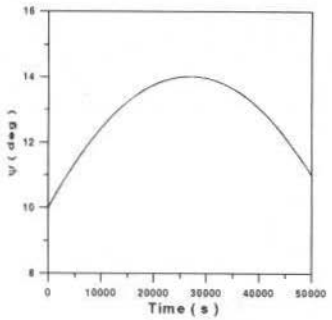


Fig. 3b Evolution of Euler Angle ψ

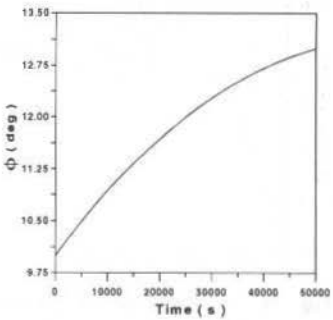


Fig. 3c Evolution of Euler Angle φ

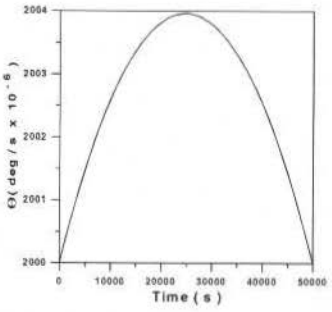


Fig. 4a Evolution of Rate variation $\dot{\theta}$

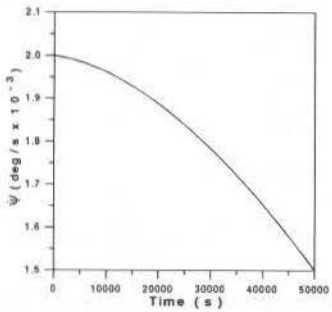


Fig. 4b Evolution of rate variation $\dot{\psi}$

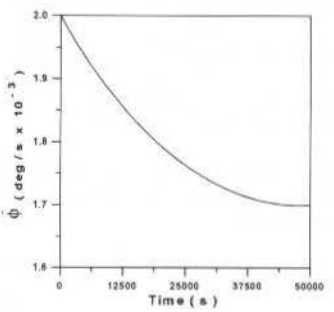


Fig. 4c Evolution of rate variation $\dot{\varphi}$

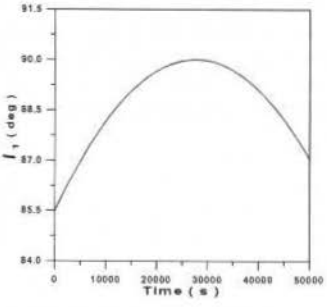


Fig. 5a Evolution of the Andoyer variable ℓ_1

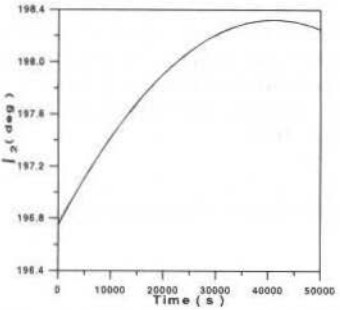


Fig. 5b Evolution of the Andoyer variable ℓ_2

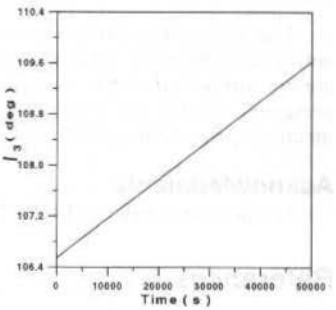


Fig. 5c Evolution of the Andoyer variable ℓ_3

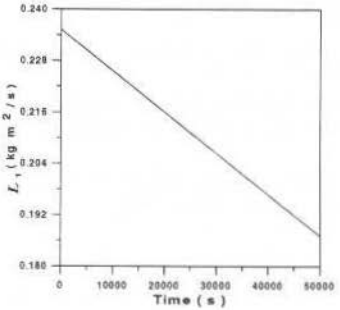


Fig. 6a Evolution of the Andoyer variable L_1

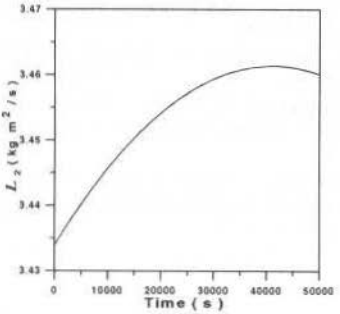


Fig. 6b Evolution of the Andoyer variable L_2

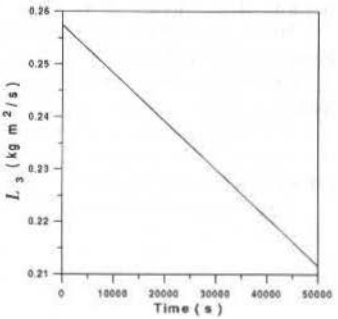


Fig. 6c Evolution of the Andoyer variable L_3

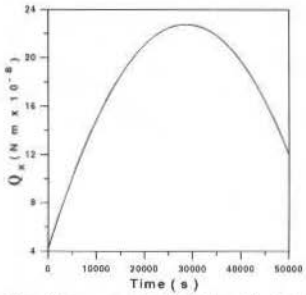


Fig. 7a Evolution of control variable Q_x

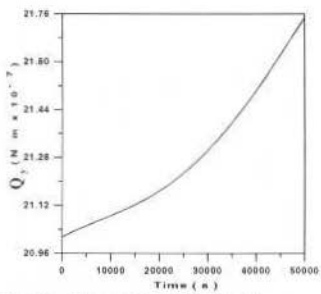


Fig. 7b Evolution of control variable Q_y

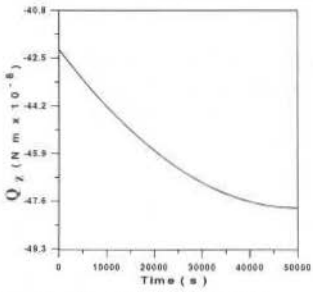


Fig. 7c Evolution of control variable Q_z

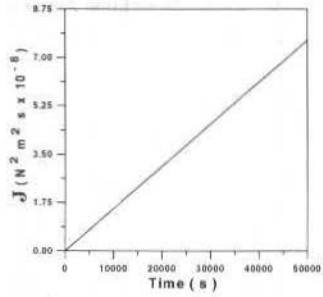


Fig. 8 Evolution of the performance index J

Conclusion

The first order solution presented here for the optimal maneuvers for artificial satellite's attitude corrections shows that the perturbations due to the gravity gradient torque and to the propulsive system are uncoupled and that is possible to establish an optimal control law for artificial satellite's attitude. The presented results are similar to the solution determined for orbit corrections maneuvers made with identical propulsion system (Da Silva, 1999).

Acknowledments

The authors thank to PROPP-UNESP for its financial support.

References

Zanardi, M. C., 1986, "Study of terms of Coupling between Rotational and Translational Motions", *Cel. Mech.*, 39, pp 147-158.

Deprit, A., 1970, "Free Rotation of a Rigid Body Studies in the Phase Plane", *Am. J. of Physics*, Vol 35, pp. 424-457.

Zanardi, M. C.; Moraes, R. V., 1994, "Effects of Solar Radiation Torque on Satellite Spin and Attitude", *J. Braz. Soc. Mech. Sci.*, Vol XV, Special Issue, pp.532-535.

Pontryagin, L. S.; Boltyanskii ,V. G.; Gamkrelidze ,R. V.; Mishchenko, E. F., 1962, "*The Mathematical Theory of Optimal Processes*", John Wiley and Sons.

Santos, R. M. K.; Da Silva Fernandes, S.; Zanardi, M. C., 1999, "Correções Ótimas de Atitude de Satélites Artificiais, *Applied Mechanics in the Americas", Proceeding of Sixth Pan-American Congress of Applied Mechanics*, Rio de Janeiro, Vol. 8, pp 1263-1266.

Da Silva Fernandes, S., 1995, "Optimum Low - Thrust Limited Power Transfers Between Neighboring Elliptic Non - Equatorial Orbits in a Non - Central Gravity Field ", *Acta Astronautica*, Vol 35, pp. 763- 770.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 449-457

ISSN 0100-7386
Printed in Brazil

Simulation Based Concept for Low Cost Attitude Operations of Equator-s

Uwe Feucht
Ralf Faller
Wolfgang Grimm
DLR Oberpfaffenhofen
German Space Operations Center
82234 Wessling, Germany
uwe.feucht@dlr.de

Abstract

This Paper describes the attitude operations concept for the scientific satellite Equator-S. The original strategy requires attitude determination and commanding capability once per orbit to operate the onboard algorithm.
Due to operational problems and in order to reduce ground station time and operator workload a different approach was implemented by using dedicated flight dynamics simulation and attitude determination algorithms.
An object oriented approach for the flight dynamics simulation had been developed for mission analysis and preparation. The capabilities of this system turned out to be the key for low cost attitude operations. Its performance had been validated to be precise enough to predict the attitude dynamics for 4 to 6 days.
In this paper the results of this more offline-oriented low cost operations strategy are presented and it is shown how precise the attitude control requirements could be met. It is demonstrated how operational resources can be saved by efforts in mission preparations and sophisticated flight dynamics systems even and especially for small satellite missions.
Keywords: *Attitude Dynamics, Flight Dynamics Simulation, Object Oriented Modeling*

Introduction

Equator-S is a scientific satellite for the investigation of the magnetosphere of the earth under the responsibility of the Max-Planck-Institute fuer Extraterrestrische Physik (MPE) in Garching, Germany. Mission Operations are performed by DLR's German Space Operations Center. The spacecraft has been launched on Dec. 4, 1997 into a final highly eccentric 500 x 65000 km orbit where it was spin-stabilised with magnetic torquers to control both spin rate and spin axis direction. The torquing maneuvers took place within 2.5 earth radii altitude around perigee.

After separation from the Ariane-4 launcher and a kick motor firing to reach the final orbit the main attitude maneuvers were the changes of the spin axis attitude from nearly parallel to the ecliptic up to 90°. There is a constraint for the sun-aspect-angle (SAA) of the solar panels mounted on the surface of the cylindrical spacecraft. Therefore it is urgent to control not only the final spin axis orientation but also the trajectory from the initial to the final attitude.

For this process the often used Shigehara algorithm had been implemented which uses the actual attitude, a selectable target attitude, orbit information and a model of the earth's magnetic field. By reducing the error in the angular momentum using an asymptotic stability criteria a switching function for the magnetic coils for periods around perigee is derived (Feucht, et al., 1997).

Due to problems with the onboard main processors the routine operational phase of the spacecraft ended on May 1st 1998 and could not be resumed so far.

All magnetic torquing is only possible within a range of 2.5 earth radii around perigee. Following the sun-aspect-angle requirements the satellite's spin axis must be erected to stay perpendicular to the orbital plane (which is in fact almost parallel to the equatorial plane). Only this attitude guarantees sufficient power supply by the solar cells independently from the season. This major attitude maneuver of the mission is performed with the aid of a ring coil aligned parallel to the spin axis. See Fig.1 for a geometry overview.

Initially the spin axis is closely parallel to the ecliptic plane with the SAA between the spin axis and the sun direction of approx. 74° (optimum would be 75° with a tolerance band of +/- 25°).

Due to the motion of the sun w.r.t. the inertially fixed orientation of the spinning spacecraft an increment of 1° per day at an ecliptical latitude of the spin axis of 0° must be compensated.

But the main target of the erection process is an ecliptical latitude of 90° which makes the SAA independent from the motion of the sun.

A generic algorithm for the interaction of magnetic torquers with the earth magnetic field to control the attitude of a spinning satellite was developed by M. Shigehara (Shigehara, 1972).

To control the spin axis direction and rate the error vector **E** between the desired spin axis direction $\mathbf{k}_f$ and the actual orientation (expressed by the angular momentum **H**)

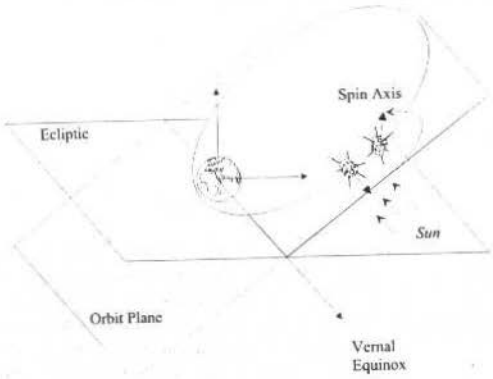


Fig. 1 Spin axis erection geometry

$$E = k_f - \frac{\mathbf{H}}{H} \tag{1}$$

should be reduced to zero.

Differentiating (1) twice with respect to time and using asymptotic stability as a sufficient condition yields after some transformations the general control criteria

$$UE\,(\mathbf{k}_B \times \boldsymbol{B}) \geq 0 \tag{2}$$

with U as the coil polarity and $\mathbf{k}_B$ as the spin axis vector in body-coordinates.

The control of the spin axis direction is then realised by the following switching function for the torquers:

$$s = (\, I_S \omega\, k - H\,)(\, m \times B\,) \tag{3}$$

with $\mathbf{I}_s$ as the tensor of principle moment of inertia around the spin axis, ω as the spin rate **m** the effective magnetic moment perpendicular to the spin axis and **B** as the vector of the earth magnetic field.

I.e. this control law which follows closely Shigehara requires the actual and the desired attitude of the spin axis, orbit information and a model of the earth magnetic field to derive a switching function for the magnetic coils for periods around perigee.

The key for making effective use of this strategy is the selection of adequate target orientations for each step of the torquing process (i.e. each perigee crossing).

The main opposite constraints are a small erection time, the effective and power-saving use of the magnetic torquers, and finally the sun aspect angle between the spacecraft spin axis and the axis satellite - sun.

Simulation System

Under these premises the key element for designing the attitude ground system baselines is a simulation and optimisation tool (Enderle, et al., 1996).

In contrary to previous missions where Fortran- or C-code was directly generated for modeling parts of the satellite's hardware and software the Equator-S mission is using a new object oriented modeling approach.

In order to use the knowledge and expertise in modeling and simulation acquired inside DLR but outside GSOC the choice was to use the Dymola (Elmqvist, 1993) environment for the modeling and simulation tasks of Equator-S.

Satellites should be described in a notation close to physics. The most natural way of modeling interacting physical systems is to describe them as individual physical objects connected according to their physical energy flow interaction. This has the advantage over modeling physical systems via signal flows or input-output block diagrams as conveniently used for controller modeling (e.g. within Simulink). I.e. that mathematical equations of the overall system have not to be ordered beforehand.

The step from the simulation to an operational software could be realized by a feature of Dymola which allows to generate standard code extracts (e.g. Fortran, C) from the simulation program. Even the complete simulation code can be extracted.

Under aspects of operational reliability and testability only the differential equations were extracted in Fortran-77 code and used for the operational software in a corresponding environment. Thus already approved modules (e.g. numeric integrator, geomagnetic field and orbit state vector predictors) could be used in order to minimize the effort in testing and maintenance.

The final design was flexible to adaptations without the need of a complete module and integration retest.

The main operational output were the switching commands for the magnetic torquer (on/off/polarity) and the onboard antennas in a time-tag format.

The commands were stored in an ASCII file which could be directly used by the command system.

Attitude Determination

The attitude solution file was generated for the EQUATOR-S science teams, as input for the spin axis erection torquing strategy, and for spacecraft spin axis surveillance purposes. It comprises time information and the spin axis direction parameterized in right ascension and declination, related to the Earth Mean Equator and Equinox of epoch J2000.0 (EME2000) coordinate system.

Due to spacecraft mass memory capacity constraints only 20 min of attitude determination relevant sensor telemetry data were available per orbit. Therefore attitude determination was performed in two 10 min time intervals, before torquing begin (40 to 30 min before perigee) and after torquing end (30 to 40 min after perigee); the respective spacecraft distances were 2.1 to 2.4 Earth radii.

The sun sensor outputs yield the sun aspect angle, i.e. the angle between spin axis and sun vector. The magnetometer measures the magnetic field vector outside the spacecraft, the x-component of the measured magnetic field unit vector yields the angle between spin axis and Earth magnetic field vector. The Earth magnetic field model used is the IGRF 1996 (Barton, 1996).

Considering the limited attitude sensor information (only two measurements to determine two attitude angles) and the inaccuracy of the Earth magnetic field model in higher altitudes, a simple deterministic single axis attitude determination algorithm was sufficient. Here the algorithm according to Grubin (Grubin, 1997) was used. The intersection of two conical surfaces yields two attitude solution vectors, and the real attitude is chosen either from an a priori estimate or from a check on the time evolution of the solutions.

The accuracies of the SAA and the magnetometer measurements were about $\pm$ 0.1 deg resp. $\pm$ 100 nT, which translated into an attitude accuracy in RA/DEC of about $\pm$ 0.6 deg.

Attitude Operations

Originally the attitude maneuvers were planned to be realised in a 3-step procedure:

- Attitude Determination:
  Telemetry data dumps were performed around the apogee, and the attitude determination yielded the spacecraft attitude after the last torquing maneuver which nominally stays inertially fixed up to the next perigee.
- *Simulation:*
  With a planned spin axis orientation as the other input the application of the Shigehara algorithm yields the switching times of the magnetic coils (Feucht, et al., 1997).
- *Commanding:*
  Before the spacecraft enters a distance of 2.5 earth radii (where the torquing maneuvers start to become efficient) these coil switching times are sent to the spacecraft as time-tagged telecommands.

This procedure was required to be used once for each orbit.

The crucial parameters for this algorithm are the input values of the actual and the planned spin axis orientation. The desired attitude is the result of a planning process (Feucht, et al., 1997) whereas the real attitude is depending from actual telemetry measurements.

The only direct attitude parameter appearing in the telemetry is the SAA measured by the sun sensor. It increases by 1°/day due to the relative motion of the sun.

Realtime telemetry cannot be received precisely at the times of perigee crossing due to the characteristics of the highly eccentric orbit.

The torquing software on ground requires the final values of the past torquing maneuver as a main input. This input can be delivered by online telemetry during phases where the coils are already switched off but the magnetic field is still strong enough to provide sufficiently precise magnetometer measurements. This leads to an earlier than planned switching off of the coils. The loss in torquing efficiency turned out to be neglectable.

The other possibility to get this data would be the storage of the magnetometer measurements by the onboard mass memory. The following dump must then occur early enough to run the ground torquing software before the next perigee requires new torquing commands.

These ground-generated time-tagged telecommands control the switching times of the onboard coils during the next perigee crossing.

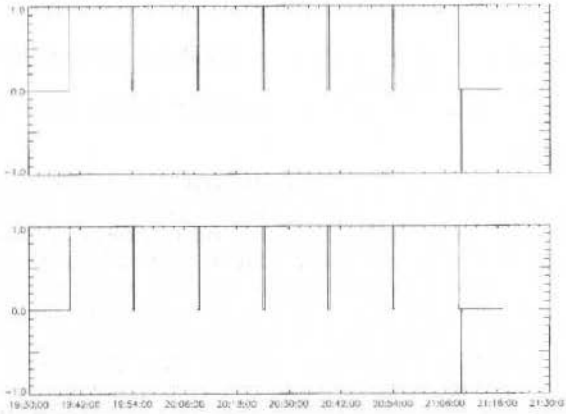


Fig. 2 Coil polarities during an early torquing

At the beginning of the torquing process with an ecliptical latitude of the spin axis close to 0° there is no change in the coil polarity necessary. The current is simply switched on and off at +/- 2.1 earth radii, except for some short periods. These non-torquing times around perigee were regarded necessary to enable precise scientific magnetometer measurements. Later during the mission it turned out that this approach did not work satisfactory.

Figure 2 shows the status of the 2 coils, the short time with negative polarity after the end of the maneuver is required for demagnetisation. Otherwise the some remanent magnetism of the coils would disturb the scientific measurement.

Though for the first 30° of spin axis erection relative to the ecliptic plane the ground algorithm only has to determine the sign of the coil current and the begin and end times of each torquing action.

The rarely available telemetry from the sun sensor is shown in Fig. 3.

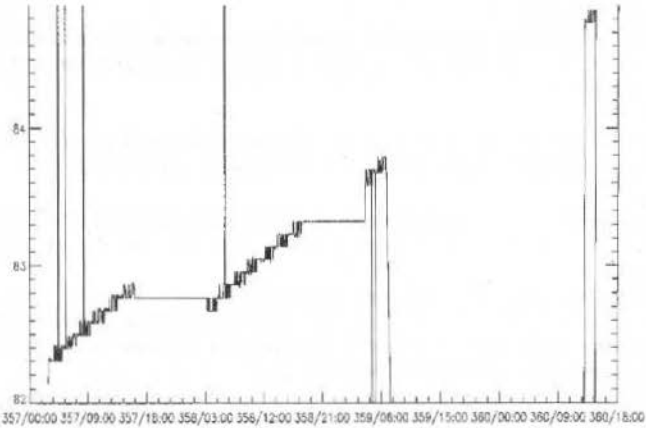


Fig. 3 SAA for perigees #34 to #37

Two longer and two shorter periods of sun-aspect-angle measurements can be identified. The linear change in the SAA is due to the relative motion of the sun around the spacecraft system. In between there are the non-measurement periods around the 4 regarded perigees including the torquing activities. In the center of the graph (perigee #35) the efficiency of the torquing maneuver (resulting in a reduction of the SAA) is almost zero, in fact during this perigee crossing the coils were switched on only for a short period.

The operational problems during the first weeks made it impossible to have access to the latest sun sensor and magnetometer measurements, i.e. for the preparation of the next torquing commands the actual orientation of the spacecraft was not known from telemetry for most of the perigee crossings.

As described above this was not important for the first weeks of the erection process because once switched on there was no change in the coil polarity sign required.

The determination of the sign at that time was only roughly dependant from the actual attitude and the on/off times for the torquing maneuver could be derived from orbit calculations.

This easy situation turned out to become more challenging as the spin axis was moved out of the ecliptical plane higher than 30°.

Now it became crucial for the efficiency of the torquing maneuvers to work with a varying coil polarity during each perigee crossing.

Remaining operational problems, an increasing demand for mass memory resources by the scientific instruments and further approaches to save ground station time made it impossible to follow the originally planned strategy of using actual attitude determination data for the generation of torquing commands.

It became a stringent requirement to predict the torquing activities for usually 4 days.

I.e. the latest attitude input data for the ground torquing algorithm could be as old as 96h or more than 4 orbits. Since the Shigehara based algorithm requires the actual spin axis orientation a precise flight dynamics simulation facility became an urgent part of the ground operation system to meet the challenges of this changed operational environment.

Due to these different operational and onboard problems it turned out that actual telemetry was not constantly available for the required times.

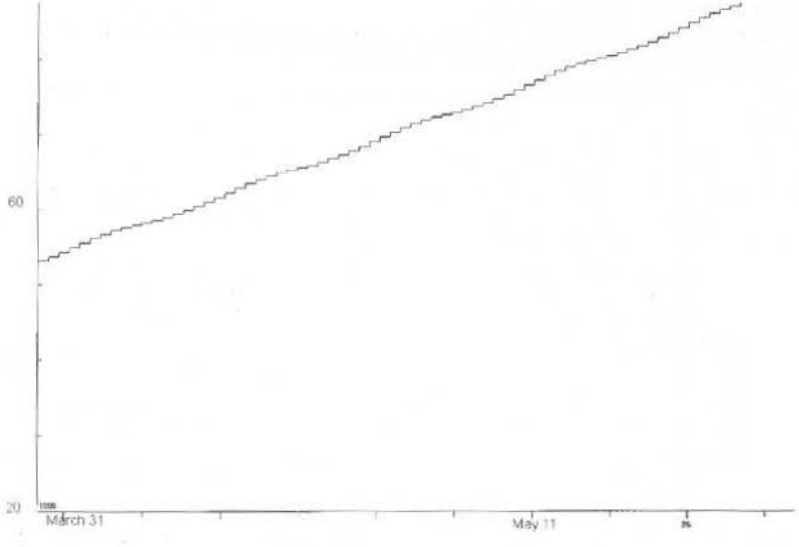


Fig. 4 Ecliptical latitude

Same difficulties lead to a lack of commanding capabilities with the result that also not before each perigee crossing the coil switching commands could be sent properly.

Thus the (sometimes) contradictory requirements of both erecting the spin axis and keeping the SAA within its limits could only be followed by closely observing the simulated satellite dynamics.

The higher the ecliptical latitude was the more precise the simulation had to meet the real spacecraft situation to generate the correct commands. Figures 4 and 5 show the long-term simulation for the erection from 53° close to the final 90° within 2 months.

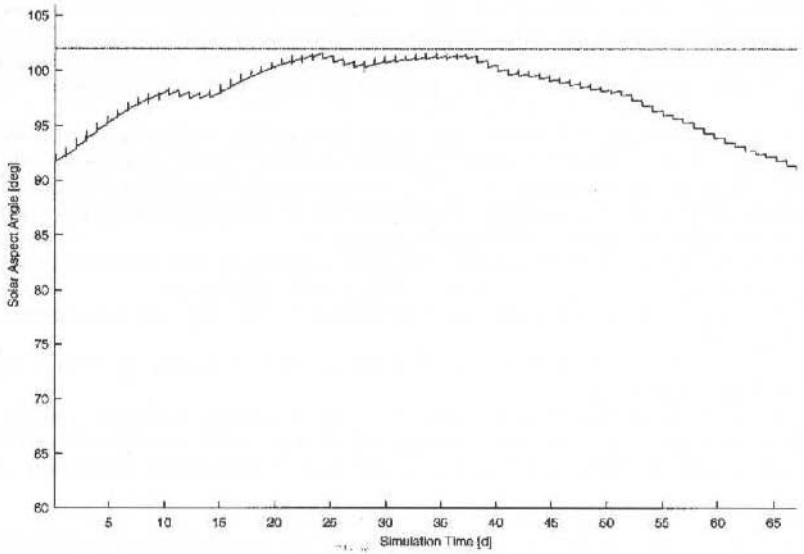


Fig. 5 SAA

Figure 4 gives the ecliptical latitude of the spin axis for the simulated last 2 months of the erection process. Each step represents a perigee crossing.

Figure 5 shows the development of the SAA for the same period. The upper line is the limit of 102° which would not be reached in this scenario.

The attitude trajectory can be visualised in equatorial inertial coordinates in Fig. 6. It shows the typical behaviour of curving up to the ecliptical north pole.

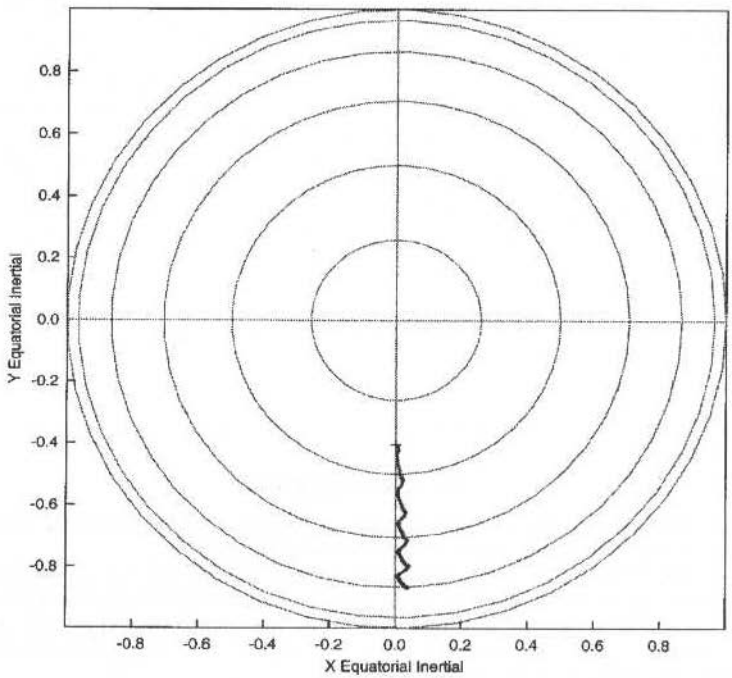


Fig. 6 Attitude trajectory

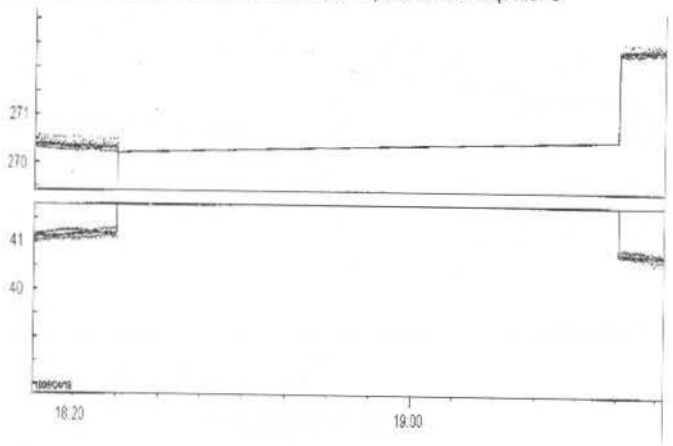


Fig. 7 Right ascension and declination of the spin axis in the equatorial inertial system

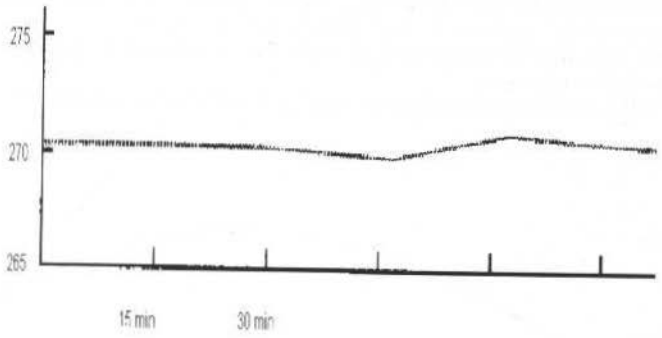


Fig. 8 Simulated right ascension

To compare the simulated satellite dynamics with the behaviour of the real satellite a typical perigee crossing of April 18, 1998 (perigee #158) is taken as an example.

Figure 7 shows the result of the attitude determination system before and after the coil activation. The periods in between without usable data result from mass memory constraints.

The equivalent change in the simulated right ascension can be seen in Fig. 8.

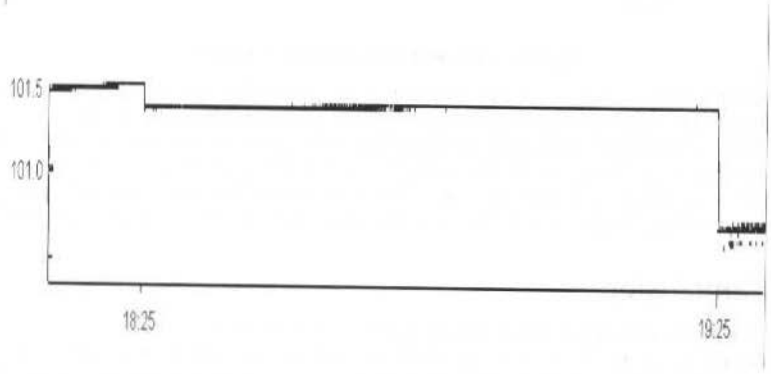


Fig. 9 SAA from telemetry

A comparison between reality and simulation for the SAA show Fig. 9 and 10 for the same torquing maneuver.

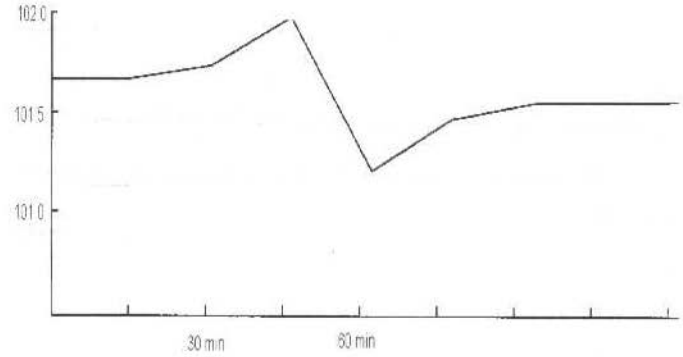


Fig. 10 Simulated SAA

Also for the SAA there are no measurements available for about 1 hour around perigee.

Another failure of the second and last onboard main processor put a (hopefully preliminary) end to the Equator-S mission during the night from April 30 to 1st of May 1998 (Melzer, et al., 1998).

This event triggered the (up to now) last analysis of the satellite's attitude dynamics. Finally the development of the SAA had to be investigated under the constraint that no more torquing maneuvers were possible in the following weeks.

Figure 11 gives the development of the SAA starting 1st of May for the next 3 months:

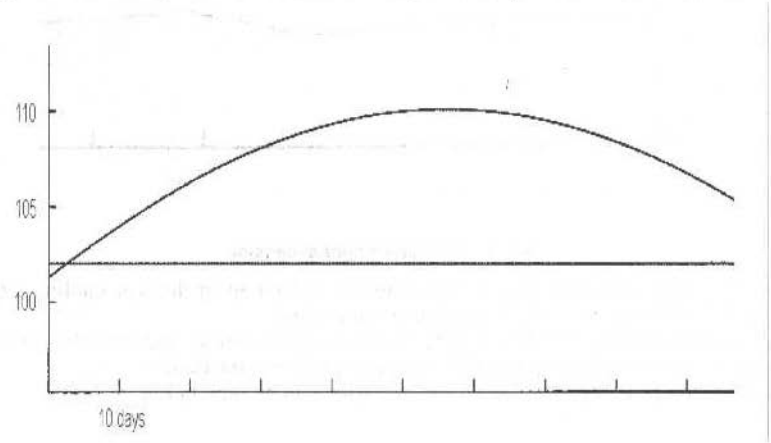


Fig. 11 SAA development after 1st of May

At that time the spin axis had already reached an ecliptical latitude of 70°. Thus the effect of the seasonal variation in the sun incidence angle on the spacecraft was only a minor one.

The maximum of 110° SAA represents no significant danger to the spacecraft or the experiments, this value is reached only for a short period, after June 22 the SAA is decreasing again. The required limit for the SAA of 102° is violated for more than 3 months, nevertheless the required low energy level onboard due to no experimental nor torquing activities would make a survival of the spacecraft possible.

Analysis, Conclusion

An analysis of the simulation accuracy yields the following results:

- Simulations of up to 1 week simulated time showed no significant orbit deviations. I.e. the influence of the orbit error on both the coil switching times and the derivation of the magnetic field vector is neglectable and far within the measurement accuracy e.g. of the magnetometer.
- The attitude dynamics, represented by its state variables and derivated parameters like the SAA showed a maximum error of approx. 1.5°. This is slightly higher than the measurement accuracy

of 1° and is reached at times of coil activity. The reasons are deviations in the modeling of the magnetic fields (both from the satellite and from the earth) counteracting with the coils.

Summary: The torquing performance would not have been better if the originally planned strategy of calculating the coil switching times once per orbit before each perigee would have been possible:

- The performance of the attitude dynamics simulation made it possible to follow the planned erection process despite different onboard and ground problems.
- With the original strategy the mission would have come into serious trouble already in January 1998 when the SAA came close to its upper limit for the first time.
- During the routine phase of the mission the performance of the simulation environment made it possible to resign from most of the scheduled ground station contacts and thus save remarkable operation costs.

Even for so called small satellites a sophisticated flight dynamics simulation system is worth the money. It can save mission costs and probably the mission itself.

Today object oriented simulation approaches make such developments possible. They can be easily reused for other missions.

References

Feucht U., Enderle W., Moormann D., 1997, "Equator-S Attitude Dynamics Simulation", *12th International Symposium on Space Flight Dynamics*, Darmstadt.

Shigehara M., 1972, "Geomagnetic Attitude Control of an Axissymetric Spinning Satellite", *J. Spacecraft*, Vol.9.

Enderle W., Feucht U., Moormann D., 1996, "Attitude Control Simulation for Equator-S using Object Oriented Modeling", *3rd International Conference on Spacecraft Guidance Navigation and Control Systems*, Noordwijk.

Barton C. E., 1996, "Revision of International Geomagnetic Reference Field Released"; *EOS Trans. 77 # 16.*

Grubin, C., 1977, "Simple Algorithm for Intersecting two Conical Surfaces", *J. Spacecraft, Vol. 14.*

Elmqvist H., 1993, "Object Oriented Modeling and Automatic Formula Manipulation in Dymola", *Scandinavian Simulation Society SIMS'93*, Kongsberg.

Melzer F., Hoefner H., 1998, "Equator-S Failure Report", *Max-Planck-Institut für extraterrestrische Physik*, Garching.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 458-464

ISSN 0100-7386
Printed in Brazil

Momentum Bias Systems with Flexible Appendages Stability Considerations

Maurizio Parisse
Carlo Arduini
Dipartimento Aerospaziale
Università Degli Studi di Roma "La Sapienza"
Via Salaria 851, 00138 Roma Italia
parisse@cralpha.psm.uniroma1.it

Abstract

The momentum bias spacecraft belongs to the wide class of the dual spin systems whole stability is ruled by the famous criterion stated by Iorillo. According to such a criterion, every energy dissipation located in the rotor has destabilizing effects whilst the energy dissipation occurring on the platform is stabilizing. The consequence is the violation of the major axis rule and the possibility for prolate systems of stable spin, provided that the dissipation rate of the platform is greater than that of the rotor, according to a certain ratio. As far as the momentum bias systems are concerned, the rotor has almost no dissipation and since this has a stabilizing effect on the platform, a nutation damper is not usually necessary. The presence of elastic appendages rigidly attached to the platform causes the rigid dynamics to be coupled with the elastic vibrations and procedure an alteration of the stability domains. This paper is devoted to the analysis of the dynamics and stability of a momentum bias systems equipped with elastic qpppendages. The stability conditions of the linearized system are derived using the Sylvester's criterion. They emphasize the importance of both the transversal inertia of the spacecraft and the angular momentum of the wheel. The system under examination very well resembles the real momentum bias system provided with solar arrays of huge dimensions for which the above conclusions are directly applicable.

Introduction

The momentum bias architecture is rather wide-spread, the system is tested and its characteristics well known. It really is an evolution of the simpler spinner of which it exploits the gyroscopic stiffness with the extra possibility of a 3-axis stabilization of the platform. For such systems, the energy dissipation plays a very crucial role.

According to the famous criterion by Iorillo, stability depends on the ratio between the dissipation rate on the rotor (destabilizing) an the dissipation rate on the platform (stabilizing). For the momentum bias system, the rotor can be assumed nondissipative so that, usually, a nutation damper is not necessary.

Although founded on heuristic bases, the Iorillo's criterion has never been contradicted by reality. When, nevertheless, it is no longer possible to model the spacecraft with two rigid bodies in relative motion duo to the presence of appendages, the criterion becomes insufficient. This is just what happens when, for instance, the momentum bias is provided with large solar array to afford the severe power demand for the instrumental package; in this case the modelling can not leave out of consideration the presence of additional masses with the relative degrees of freedom, however, the increased size of the model is necessary for a more accurate stability analysis.

The Model

The system under examination has been modeled with a triinertial rigid boy R which carries a rotor W whose simmetry axis has a generic orientation.

A principal inertia body frame with the origin located in the center of mass of $R+W$ has been adopted. The apendages are roughly modeled by the masses m attached at the tips of two massless bars connected to R by means of elastic hinges of stiffness k.

The position of the masses m with respect to the boy frame is defined by the angles ϑ_{1x}, ϑ_{1z}, ϑ_{2x}, ϑ_{1z} which are supposed to be <<1. The rods in the rest position are aligned along the y axis. The unit vector $\vec{a}$ indicates the direction of the rotor rotations axis while ω is the absolute angular velocity of the body frame (Fig. 1).

The mathematical model has been derived with a lagrangian approach. The position of the masses is defined by the vectors:

$$P_1 = \begin{cases} -\ell\vartheta_{1z} \\ \ell - y_0 \\ \ell\vartheta_{1x} \end{cases} \qquad P_2 = \begin{cases} -\ell\vartheta_{2z} \\ -\ell - y_0 \\ -\ell\vartheta_{2x} \end{cases}$$

so that

$$\vec{V}_1 = \dot{\vec{P}}_1 + \vec{\omega} \times \vec{P}_1 \qquad \vec{V}_2 = \dot{\vec{P}}_2 + \vec{\omega} \times \vec{P}_2$$

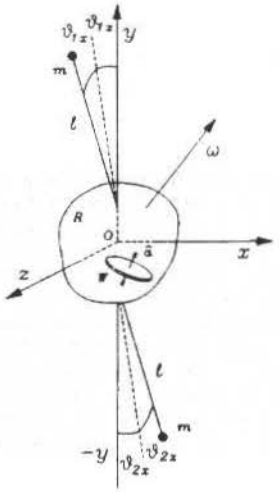


Fig. 1

If $\vec{r}$ indicates the position of a generic point o the subsystem $R + W$ with respect to O, $\vec{r}_W$ the position of a generic point of the rotor with respect to its own center of mass and $\vec{\omega}_r = \hat{a} w_r$ the relative angular velocity of W with respect to R, the kinetic energy can be written:

$$2T = \int_R (\vec{\omega} \times \vec{r}\,).(\vec{\omega} \times \vec{r}\,)dm + \int_W^R (\vec{\omega} \times \vec{r} + \vec{\omega}_r \times \vec{r}_W\,).(\vec{\omega} \times \vec{r} + \vec{\omega}_r \times \vec{r}_W\,)dm +$$

$$+ m(\vec{V}_1.\vec{V}_1 + \vec{V}_2.\vec{V}_2) = \vec{\omega}\vec{\vec{J}}\vec{\omega} + \omega_r^2 I_s + 2\vec{\omega}.\hat{a} I_s \omega_r + m(V_1^2 + V_2^2)$$

where $\vec{\vec{J}}$ is the principal inertia tensor of the subsystem $R+W$, computed respect to O, and I_s the axial inertia moment of the rotor. Taking into account the potenzial energy stored in the elastic hinges, the system lagrangian is:

$$2L = \vec{\omega}\vec{\vec{J}}\vec{\omega} + \omega_r^2 I_s + 2\vec{\omega}.\hat{a} I_s \omega_r + m(V_1^2 + V_2^2) - k(\vartheta_{1x}^2 + \vartheta_{1x}^2 + \vartheta_{2x}^2 + \vartheta_{2x}^2)$$

By observing that the generalized coordinates set is composed by quasi coordinates and true coordinates, the motion equation are derived by applying the classical formalism:

$$\begin{cases} \dfrac{d}{dt}\dfrac{\partial L}{\partial \{\omega\}} + [\omega]\dfrac{\partial L}{\partial \{\omega\}} = 0 \\ \dfrac{d}{dt}\dfrac{\partial L}{\partial \{\dot{\vartheta}\}} + [\omega]\dfrac{\partial L}{\partial \{\vartheta\}} = 0 \end{cases}$$

The next step is the linearization of the above equations in order to investigate the stability domains. With a general configuration such as that sketched in Fig.1, the equilibrium positions around which to linearize can be found, according to Hughes (Hughes, 1986), by imposing the condition $\dot{\omega}$ = const and by exploiting the first integral of the angular momentum. Nevertheless, we are here interested in the classical flight configurations so that, with reference to the rotor spin axis, we suppose it aligned with the y axis together with a slightly perturbed spin v of the platform around the same axis. With these hypotheses the absolute angular velocity can be written:

$$\{\omega\}=\{\dot{\alpha}\}+[U-\tilde{\alpha}]\{v\} \qquad \{v\}=\begin{Bmatrix}0\\ v\\ 0\end{Bmatrix}$$

where U is the identity matrix and $\{\alpha\}=\{\alpha_1\,\alpha_2\,\alpha_3\}$ represents the (very small) angular displacement of the body frame with respect to the fixed frame, in the Euler sequence 1-2-3. By substituting the linearized expression of ω we fin the canonical system:

$$M\ddot{q}+G\dot{q}+Kq=0$$

where

$$q^t=\{\alpha_1\,\alpha_3\,\vartheta_{1x}\,\vartheta_{1z}\,\vartheta_{2x}\,\vartheta_{2z}\}$$

$$M=\begin{bmatrix} \bar{I}_1 & 0 & m\ell l & 0 & -m\ell l & 0\\ 0 & \bar{I}_3 & 0 & m\ell l & 0 & -m\ell l\\ m\ell l & 0 & m\ell^2 & 0 & 0 & 0\\ 0 & m\ell l & 0 & m\ell^2 & 0 & 0\\ -m\ell l & 0 & 0 & 0 & m\ell^2 & 0\\ 0 & -m\ell l & 0 & 0 & 0 & m\ell^2 \end{bmatrix}$$

$$G=\begin{bmatrix} 0 & (\bar{I}_3-I_2+\bar{I}_1-\frac{h_r}{v})v & 0 & 2m\ell lv & 0 & 2m\ell lv\\ -(\bar{I}_3-I_2+\bar{I}_1-\frac{h_r}{v})v & 0 & -2m\ell lv & 0 & -2m\ell lv & 0\\ 0 & -2m\ell lv & 0 & 2m\ell lv & 0 & 0\\ -2m\ell lv & 0 & -2m\ell lv & 0 & 0 & 0\\ 0 & -2m\ell lv & 0 & 0 & 0 & 2m\ell lv\\ -2m\ell lv & 0 & 0 & 0 & -2m\ell lv & 0 \end{bmatrix}$$

$$K=\begin{bmatrix} (I_2-\bar{I}_3+\frac{h_r}{v})v^2 & 0 & -m\ell lv^2 & 0 & -m\ell lv^2 & 0\\ 0 & (I_2-\bar{I}_1+\frac{h_r}{v})v^2 & 0 & -m\ell lv^2 & 0 & -m\ell lv^2\\ -m\ell lv^2 & 0 & k-m\ell^2lv^2 & 0 & 0 & 0\\ 0 & -m\ell lv^2 & 0 & k-m\ell lv^2 & 0 & 0\\ -m\ell lv^2 & 0 & 0 & 0 & k-m\ell^2lv^2 & 0\\ 0 & -m\ell lv^2 & 0 & 0 & 0 & k-m\ell^2lv^2 \end{bmatrix}$$

and the following positions have been made: $l=y_0+l, h_r=I_s\omega_r$ and $\bar{I}_i=I_i+2ml^2$; y_0 is the position of the hinge along the y axis.

Stability Analysis

The stability of the linear system in the canonical form is ensured by the positive definiteness of the stiffness matrix K. This, in turn, is strictly related to the satisfaction of the Sylvester criterions which provides the stability conditions:

$$I_2 > \bar{I}_3 - \frac{h_r}{v}$$

$$I_2 > \bar{I}_1 - \frac{h_r}{v}$$

$$\Omega^2 = \frac{k}{ml^2} > \left[1 + \frac{2ml^2}{I_2 - \bar{I}_3 + \frac{h_r}{v}} \right] v^2$$

$$\Omega^2 = \frac{k}{ml^2} > \left[1 + \frac{2ml^2}{I_2 - \bar{I}_1 + \frac{h_r}{v}} \right] v^2$$

The original six conditions shrink to four because two of them are included in the corresponding stronger ones. However, the positive definiteness of K (static stability), is a sufficient but not necessary condition. The presence at the term $G\dot{q}$ can actually add stable configurations which are statically unstable but gyroscopically stabilized. Nevertheless, these stable regions are usually cancelled by damping always present in the real systems.

The first two conditions state that if h_r and v, are suitably chosen. With different signs, stability is no longer ensured simply by a spin around the maximum inertia axis but this inertia must be greater than a certain value depending on the wheel characateristics. In other words the term h_r/v represents a fictitious inertia whose sign could stabilize or destabilize the satellite spin.

The last two conditions concern the additional masses m and state the requirement that the string stiffness are designed so that the elastic forces could counteract the centrifugal forces.

These conditions can be arrange in a different way in order to highlight the importance of the transversal inertia margin:

$$I_2 - \bar{I}_3 + \frac{h_r}{v} > \frac{2ml^2}{\frac{\Omega^2}{v^2} - 1}$$

$$I_2 - \bar{I}_1 + \frac{h_r}{v} > \frac{2ml^2}{\frac{\Omega^2}{v^2} - 1}$$

That is, for a prescribed ratio $\Omega^2 / v^2 > 1$, we have stability if the transversal inertia, including the fictitious one, is greater than a certain value.

From the first two stability conditions it is possible to write:

$$I_2 = I_3 + 2ml^2 - \frac{h_r}{v} + \Delta_1$$

$$I_2 = I_1 + 2ml^2 - \frac{h_r}{v} + \Delta_2$$

where Δ_1 and Δ_2 are just the transversal inertia margins, small margins require large value of the spring stiffness k and viceversa (Fig. 2). Of course, the most demanding condition, with the lower Δ, applies.

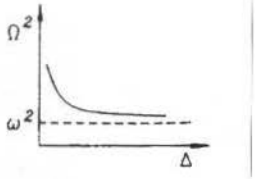


Fig. 2

For more realistic analysis let us introduce the damping matrix D to simulate the energy dissipation of whatever nature occurring in the appendages or in the main body.

The canonical systems becomes:

$$M\ddot{q} + (G + D)\dot{q} + Kq = 0$$

where the matrix D has at least one element $\neq 0$ on the main diagonal (pervasive damping).

With the position $u^t = \{q\ \dot{q}\}$ the system can be written as:

$$\dot{u} = Au$$

where

$$A = \begin{bmatrix} 0 & U \\ -M^{-1}K & -M^{-1}(G+D) \end{bmatrix}$$

The iegenanalysis of the matrix A and the tracing of the real part of the computed eigenvalues allows the definition of the stability domains in the k_1 and k_3 plane. The classical ratios k_1 and k_3 are defined as follows:

$$k_1 = \frac{I_2 - I_3}{I_1} \qquad k_3 = \frac{I_2 - I_1}{I_3}$$

however, for the sake of simplicity, we restrict the analysis to a double axialsimmetric system for which $I_1 = I_3 = I$ so that

$$k_1 = k_3 = k_{13} = \frac{I_2}{I} - 1$$

This is equivalent to examine those systems that, due to their own inertia values, are located along the bisector of the first an third quadrant of the k_1 and k_3 plane.

In principle, it would be possible to analytically define the stability boundaries by analyising the coefficients of the secular equation of the matrix A. This task is however hard and tedious.

The results are shown in the following diagrams (Fig. 3); the numerical values of the main parameters involved in the analysis are summarized in table 1, for the others we have $v = 1$ *rad/s*, $k = 500$ *N m*, $l = 5$ *m*, $y_0 = 1$ *m*, $I = 1000$ *Kg m*^{2}.

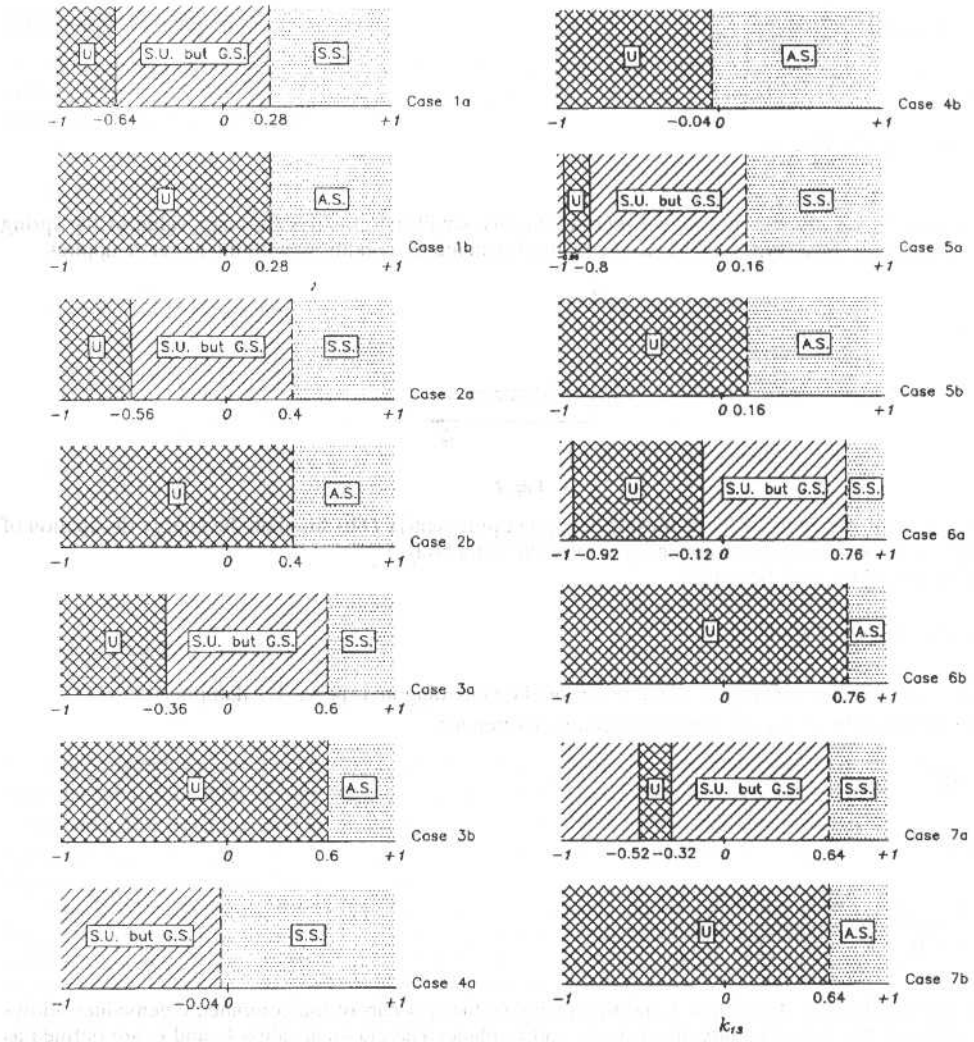


Fig. 3

Table 1 Cases investigated

Case	Mass m [Kg]	h_r [Nms]	Damping coefficient
1a	5	200	0
1b	5	200	0.1
2a	5	100	0
2b	5	100	0.1
3a	5	-100	0
3b	5	-100	0.1
4a	0.5	100	0
4b	0.5	100	0.1
5a	0.5	-100	0
5b	0.5	-100	0.1
6a	2	-600	0
6b	2	-600	0.1
7a	0.5	-600	0
7b	0.5	-600	0.1

Conclusions

As expected, the presence of the damping cancels the statically unstable (S.U) but gyroscopically stabilized (G.S.) regions which become unstable (U), while the original statically satble (SS) regions become asymptotically stable (A.S.).

The effect of h_r which provides the gyroscopic stiffness to the system, is evident, Its increase causes an enlargement of the stability area. The same effect is produced by the decrease of the m that, reducing the value of the gyroscopic force, allows, under the same conditions a less penalizing mass distribution.

Large size solar arrays with significant mass, necessary to meet high power demand, directly affect actually, the stability conditions concerning Ω^2. To satisfy these conditions, an increase of the mass causes a consequential increase of I_2.

References

Hughes, P. c., 1986, "Spacecraft Attitude Dynamics". John Wiley & Sons.

Wittenburg, J., 1977, "Dynamics of System of Rigid Bodies". B. G. Teubner. Stuttgart.

Mingori, D. L., 1969, "Effects of Energy Dissipation on the Attitude Stability of Dual-Spin Satellites". AIAA J., 7 (No. 1) 20-27.

Huseyin, K., 1978, "Vibrations and Stability of Multiple Parameter Systems". Sijthoff an Nordoff, Rockville, Md.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 465-470

ISSN 0100-7386
Printed in Brazil

An Optimal Search Algorithm for Satellite Acquisition

Pasquale Di Stasio
Glauco Di Genova
Telespazio S.p.A.
cratos@tin.it

Abstract

The Cooperative Search Algorithm (CSA) is an algorithm developed in the framework of an IRIDIUM™ task; the purpose was to provide a procedure for directing ground antennas in satellites searching; a software package for the Ground Segment was implemented together with a software testbed for the performance evaluation of the search algorithm. The subject of this paper is the description of the CSA in terms of algorithm philosophy as well as the description of the results obtained by using the CSA and the testbed software in the simulations of the nominal acquisition scenario.

Keywords: *Algorithm, Search, Antenna, Acquisition, Testbed.*

Introduction

The software package Cooperative Search Algorithm (CSA) is the implementation of an algorithm to be used to direct ground antennas in satellites searching. It was developed in the framework of an IRIDIUM task to provide a procedure both for satellites acquisition after their launcher release and for satellites re-acquisition in mission orbit after contact failure, by means of one or more ground antennas. Moreover a software testbed for the performance evaluation of the search algorithms was developed.

The IRIDIUM™ constellation

The IRIDIUM™ constellation consists of 66 satellites in circular, nearly polar orbits (with an inclination of 86.1°) of 780 kilometers altitude. The satellites are placed in six planes (11 satellites per plane) equally spaced over 180° of longitude providing worldwide coverage. Initially, the satellites (in groups of five or more satellites) have been placed in a parking orbit of approximately 460 kilometers, released by different launchers (DELTA, Proton, Long March). From there they have performed orbit raising maneuvers to either their storage orbit of approximately 650 kilometers or their mission orbit.

The Ground Segment is in contact with the constellation of satellites through a network of four Telemetry, Tracking & Command (TT&C) facilities: two ground antennas are located in northern Canada, one in Haway, and one in Egil (Island).

The algorithm was applied to a typical configuration (nominal scenario) during the simulations: two ground antennas for the acquisition of five satellites released by the launcher DELTA. As input parameters were assumed as known the initial (at the launcher separation) state vectors, the initial covariance matrix, and the antenna characteristics, that is Field Of View (FOV), maximum rate, maximum acceleration, and elevation cut-off. In addition to this scenario different cases were also considered, with a single tracking antenna, with the TT&C network in full configuration (four ground stations), and with different elevation cutoffs.

The CSA task

The task execution was based on several activities:

- Modeling of the satellite dynamics after launcher separation and in mission orbit;
- Covariance analysis and ground tracks determination for different launchers;
- Dynamical modeling of antenna pointing mechanism to the extent needed to take into account slewing rates, accelerations, and pointing constraints;
- implementation of two algorithms (multi antenna -one satellite and multi antenna - multi satellite) and evaluation of their performance.

The successful outcome of the work was strongly dependent on the possibility to reduce as much as possible, through accurate modeling the portion of sky in which to search for the satellites: the acquisition of the satellites after their launcher separation is actually a problem more complicated than the loss of the contact with the satellite in mission orbit. Apart from the information about the satellite dynamics during the first revolutions after the separation (position, velocity, sequence of maneuvers planned, attitude data, etc.), the critical data were represented by the satellites release dispersion data.

A software tool was developed and used to simulate the satellite dynamics and the link acquisition process, acting as a software testbed for the performance evaluation of the search algorithm: although this work was performed specifically to simulate the IRIDIUM^{TM} satellites acquisition by using the CSA, the software tools developed are generic enough to be used for the simulation of a number of scenarios relevant to a generic satellite mission analysis.

CSA description

CSA functional description

The CSA program generates both a binary file containing the tracking data (azimuth and elevation angles) for each single antenna in order to search the satellites released by launcher, and a text file containing the same information along with some auxiliary quantities (at least range and range rate).

The algorithm was applied to a configuration based on two antennas which followed the satellite trajectory. The adopted search strategy used the two antennas to track a point on a reference orbit moving respectively faster and slower than an actual satellite on that orbit, that is an antenna operated in *fast-scanning* mode and the other one in a *slow-scanning* mode.

The initial covariance matrix was assumed to be the same for all the satellites released. The initial sigmas (1• values) were 1600 meters, 600 meters, and 4400 meters in Height, Cross-track, Along-track (HCL) components. The velocity dispersions were 2 m/s in all the HCL components. These sigmas were inferred from the 99th percentile levels of the injection parameters, without relying on any particular hypothesis about the details of their statistical distribution (i.e. they were <u>not</u> assumed to follow a Gaussian distribution). From dispersion analysis data it resulted that the main satellite trajectory dispersions developed along the in-track direction with a rate approximately constant in time, while the cross-track and radial dispersion oscillated between bounds which were approximately constant over a few hours time span; thus the main search direction has been selected to be the along-track one.

Without relying on any particular hypothesis on the trajectory distribution apart from the obvious one that the most probable trajectories were the closest to the nominal one, the search started somewhere around the nominal trajectory at the pass start time. Since along-track errors develop both ahead of and following the nominal satellite position, then the search was performed by scanning both the trajectory arcs starting at the nominal satellite position and moving away from there forward and backward, respectively.

The angular speeds for the two antennas were determined by the need of keeping every point which moves with the satellite velocity in the antenna beam width for at least 1 sec (the time needed for auto-track acquisition). Once the boundary of the along-track dispersions have been reached, then the next major source of the uncertainty was taken into account namely the radial error. Thus the two antennas were pointed towards the nominal satellite position, but with a bias in the radial direction and the same *slow-fast scanning* mode as before was used to cover an along-track error region lying at a certain radial distance from the nominal trajectory. The radial bias was selected in such a way to overlap only slightly the already scanned region, to the extent needed to avoid gaps in the coverage of points (moving with satellites velocity) lying inside the beam width for at least 1 sec. The entire process of adding a radial bias and performing a slow, fast pair of scans was then iterated until the end of the pass for positive and negative radial biases.

The CSA output file can be logically split into pointing data segments corresponding to different radial biases. The antenna pointing angles in each pointing data segment of the CSA file allow to cover an along track region of sky (a strip) in the searching area. The search strategy adopted in the nominal scenario allowed to use the CSA program to generate files containing three complete pointing data segments within the available pass time. The first one was relevant to a strip covering the region of highest probability, that is the region around the nominal orbit. The antennas followed the satellite trajectory, scanning such an arc to take into account the along-track dispersion; then the CSA program generated other two pointing data segments corresponding to scans at lower and higher radii than the nominal orbit.

The CSA program estimated both the angular rate and the angular biases, so that ,

1) the satellite was maintained in the antenna FOV at least for the auto track acquisition time,
2) the sky region scanned by the antennas contained the most part of the along-track and radial dispersions,
3) the overlap between the strips was reduced as much as possible.

Two parameters were used in the algorithm in order to satisfy the requirements 1, 2, that is the scan duration and the FOV crossing time.

The covariance analysis showed that the uncertainty in the satellite position determination was due mainly to the along-track error, so that the scan duration is related to the amount of the along-track dispersion. The FOV crossing time is the time which a satellite passing through the antenna beam axis spends in the antenna FOV. This time has to be larger than the auto track acquisition time in order to allow even an off-axis satellite to be kept within the beam for at least the auto-track acquisition time.

The size of the cross-track and radial region effectively scanned by an antenna thus depends on the crossing time over acquisition time ratio.

CSA Performance Evaluation

To assess CSA performance, a set of software tools (testbed) has been developed whose purpose is both the automatic computation of some dynamical parameters relevant to the CSA itself, and the simulation of the acquisition process of one or several satellites from one or several ground stations in order to compute the acquisition probability for that scenario. Although this work has been performed specifically to simulate the IRIDIUM™ satellites acquisition by using a CSA, the software tools developed are generic enough to be used for the simulation of a number of scenarios relevant to a generic satellite mission analysis. In fact, the simulation software implements separately the trajectory simulation and statistical analysis, and the acquisition process simulation. In order to achieve a realistic implementation of the actual mission scenario, a statistical simulation approach was selected.

The simulation purpose is twofold. On one hand, given an externally provided CSA file containing time tagged pointing data (e.g. azimuth, elevation), it should be possible to compute the performances of this CSA file in terms of probability of acquisition and acquisition time. On the other hand, the trajectory simulation process should be used to provide some insight into the acquisition process itself, that is to produce input to the CSA such as, for example, the satellite state vector dispersions and their time evolution: this is a support function to the operational CSA, while the CSA performance analysis allows a quick assessment of the capabilities of the CSA under test.

The testbed software reads each CSA file and checks for satellite acquisition on each one of the sample trajectories generated during the Monte Carlo simulation. The number of trajectories generated may be selected depending on the desired accuracy in the determination of the acquisition probability. Azimuth and elevation data from the CSA file are interpolated separately.

The acquisition check is based on two parameters, that is the FOV and the acquisition time. The pointing error is the difference between the commanded pointing direction and the pointing direction at which the satellite can be found, evaluated at each sampling time (0.1 sec) throughout the entire pass duration of the particular sample trajectory considered. If this pointing error stays below the FOV parameters for a time interval longer than the acquisition parameter, then on that particular sample trajectory acquisition occurs. The computed statistical performance indexes are: the acquisition probability, the average acquisition epoch and the average acquisition epoch dispersion.

Simulation of the Nominal Scenario

Testbed Approach

The CSA testbed aims at modeling the satellite acquisition process in the closest possible way to the real environment: to this purpose a Monte Carlo approach has been selected. This consists of the generation of a large number of different satellite trajectories representing a statistical population containing (practically) all the possible trajectories compatible with a given set of initial dispersion data.

The fact has to be stressed that CSA does not rely on any particular hypothesis about the shape of the state vector uncertainty distribution. In fact even if in the framework of the IRIDIUM™ task a Gaussian distribution was assumed for sake of simplicity, being CSA an algorithm tailored to perform operational work it relies on the best possible approximation to the real physical environment. The truth model adopted there was a statistical distribution whose initial parameters were assumed to be consistent with the available initial 99% percentile levels at orbit injection. The initial trajectory distribution was then generated according to a Gaussian distribution, for lack of better knowledge about its shape, even if a truncated Gaussian or a triangular were better approximations; anyway the nonlinear transformation between initial and final states following from the equations of motion produces a non Gaussian distribution at later times. Use of a Monte Carlo method allows for easy implementation of any meaningful initial distribution.

Probability levels approaching the 100% figure were considered, that is the tails of the trajectory probability distribution. In order to achieve confidence levels close to 100%, also trajectories with probability of occurrence very low were considered; these trajectories are strongly dependent on the

details of the probability distribution. Meeting the 99% acquisition probability requirement meant to consider trajectories lying between the 0.5 and the 99.5 percentile level bounds.

Testbed Description

The simulation approach is based on a pipeline of five main modules to be activated in sequence, and on several support modules. Each main module reads the same simulation setup file and, when appropriate, an intermediate file produced by the module previously run.

The five main modules, named CSATEST1-5, implement a Monte Carlo simulation of the satellite acquisition process, given a CSA file containing time tagged pointing data for a number of antennas at a ground site. For each complete simulation several performance indexes related to the given CSA file can be computed. The Monte Carlo trajectory simulation and analysis (modules CSATEST1-3) is kept separate from the CSA performance evaluation (modules CSATEST4-5); thus it is sufficient to run the first three modules once and for all to produce the statistical population and then a single run of CSATEST4-5 for each proposed CSA will provide the required performance data. When several antennas are used at the same ground station, each one should correspond to a CSA file, and CSATEST4 should be run once for each antenna, while CSATEST5 computes the combined performance data for the simultaneous search from several antennas. In case of a single antenna used, then it is sufficient to run CSATEST4 only.

Nominal Scenario Setup

The scenario considered in the simulations was the nominal one with two ground antennas and five satellites released by the DELTA launcher; the initial state vectors used throughout the simulations took into account separation effects, but satellite stabilization effects were neglected. These stabilization effects were accounted for in the initial covariance matrix adopted. The characteristics of the antennas used in the satellites searching where:

- FOV (from antenna axis) = 0. 125 deg
- maximum X rate = 5.0 deg/sec
- maximum Y rate = 5.0 deg/sec
- maximum X acceleration = 5.0 deg^2/sec
- maximum Y acceleration = 5.0 deg^2/sec
- elevation cut-off = 10 deg

The acquisition strategy adopted, providing the best results in terms of acquisition probability, was obtained by running twice the CSA program: the first run generated the CSA file of the antenna operating in *slow-scanning* mode, the second one generated the CSA file of the antenna operating in *fast-scanning* mode.

Simulation Results

A number of simulations related to the nominal scenario was performed, dealing with different numbers of ground antennas at each site and with different elevation cutoffs (5 and 8 deg). Each simulation was performed by maximizing the acquisition probability during the search time. For each simulation three different search times were considered, namely, 100, 180, and 260 seconds. Results are summarized in the following tables.

Table 1 Acquisition probability (%) (Time search = 100 seconds)

	1 antenna	2 antennas	3 antennas
10 deg	69.2	77.8	89.5
8 deg	76.9	83.5	92.3
5 deg	85.9	91.7	95.8

Table 2 Acquisition probability (%) (Time search = 180 seconds)

	1 antenna	2 antennas	4 antennas
10 deg	77.1	86.5	96.6
8 deg	83.3	90.3	98.2
5 deg	90.7	95.3	99.3

Table 3 Acquisition probability (%) (Time search = 260 seconds)

	1 antenna	2 antennas	4 antennas
10 deg	78.8	94.4	95.8
8 deg	84.9	95.2	99.3
5 deg	94.1	97.3	99.7

In these simulations the first pass occurred during the 2nd orbit after release. The results related to a single antenna are very sensitive with respect to the shape of the initial probability of distribution (Gaussian in that case): this led to a larger than expected acquisition probability. This depends on the fact that for a strictly Gaussian distribution, the bulk of that distribution is contained in a relatively small volume which is the one covered by just a single antenna.

In addition to the nominal scenario, which assumed a pass whose maximum elevation was above 10 degrees, the case was considered when passes whose maximum elevation was above 5 degrees were suitable for satellite acquisition. Then the first pass occurring during the 1st orbit after release was investigated: the result obtained was that there was an acquisition probability of practically 100% within 100 seconds by using a single ground antenna, as reported in the following tables. In these simulations the maximum elevation was 8.2 deg.

Table 4 Acquisition probability (%) (Time search = 100 seconds)

Initial elevation	1 antenna	2 antennas
5 deg.	99.8	100

Table 5 Acquisition probability (%) (Time search = 180 seconds)

Initial elevation	1 antenna	2 antennas
5 deg.	100	100

Alternative search strategies

The strategy adopted was the one able to produce the best results, that is the strategy maximizing the acquisition probability. The possibility of considering alternative search strategy, feasible in principle, is constrained by some rather obvious requirements:

- The search has to cover the most probable region first in order to reduce the average acquisition time;
- The search should cover as much as possible the error region;
- The most efficient search occurs at low elevation angles since there the antenna footprint is larger;
- It is better to avoid a priori assumption about the shape of the statistical distribution of the orbit state vector errors;
- The commanded angular speed and rates do not have to exceed specified limits.

The above mentioned requirements rule out most of the random search modes trying to get exhaustive coverage of the error region (e.g. raster or spiral scans).

The constraint of having to keep a satellite in the antenna beam width for a certain time implies that the main antenna motion in a program-track mode has to be in the direction of a satellite motion, that is along-track; this significantly reduces the possibility of adopting alternative strategies.

The requirement of having to scan the most probable regions first means that the scan motion should start around a nominal position and should continue moving away from that position towards the error region boundaries: this excludes a scan starting close to the boundaries and moving inward.

The requirement of trying to avoid particular assumptions on the distribution of the orbital state errors has a physical meaning. The assumption of a Gaussian distribution was misleading at least: for the CSA there was the need to reach the boundaries of the error region, while for a Gaussian distribution the most probable region is relatively small and far from the boundaries that is the 4 sigma levels.

The only degree of freedom left is how different scan arcs are allocated to the two ground antennas. A search strategy in which the antenna motion was side-by-side, that is in which the two ground antennas motion is coupled, was adopted and the results are presented below:

Table 6 Acquisition probability (%) (Time search = 100 seconds)

Initial elevation	1 antenna	2 antennas
10 deg.		91.0

Table 7 Acquisition probability (%) (Time search = 180 seconds)

Initial elevation	1 antenna	2 antennas
10 deg.		91.2

Table 8 Acquisition probability (%) (Time search = 260 seconds)

Initial elevation	1 antenna	2 antennas
10 deg.		91.5

A comment about this result is that these simulations were performed maximizing the acquisition probability for a single antenna, since the second one just followed the first. Then, as in the single antenna scan results presented in the previous section, the acquisition probability figure is misleading since it is very sensitive with respect to the assumption of Gaussian error distribution.

As a final remark, we wish to point out that the CSA has been actually used during the actual IRIDIUM™ launches, and it performed as foreseen.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 471-484

ISSN 0100-7386
Printed in Brazil

Optimal Motion and Position Control of Nonholonomic Flexible Arm

Felipe Eduardo de la Rosa Bocanegra
Graduate School of Science and Technology , Keio University
3 – 14 – 1 Hiyoshi , Kouhoku – ku , Yokohama 223-8522 Japan
m981889@msr.st.keio.ac.jp

Tadahiro Fujio
Graduate School of Science and Technology
Keio University

Kazuo Yoshida
Faculty of Science and Technology
Department of System Design Engineering , Keio University

Abstract

A robot arm operating in such environments as spacecraft and space vehicles, attached to a space platform, station or satellite, has to show considerable ligthness. This property can be achieved by an underactuated manipulator with links possessing some flexibility. In this paper, a two-flexible-link manipulator with the second joint unactuated and therefore a nonholonomic system , is modeled and a control method is proposed to attain a desired position from an initial position. As far as this system is categorized into underactuated manipulators, an exact solution for the optimal trajectory poses considerable difficulties. In this case, to find the optimal trajectory and control input for the cost function to be minimized, we make use of the Ritz method. Through the application of Fourier Basis Algorithm, the near optimal control parameters are acquired, and the solution is approximated by solutions of some finite-dimensional systems and then those solutions converge to the optimal solution.
Keywords *: Nonholonomic system, underactuated manipulator, flexible link, Fourier Basis Algorithm.*

Introduction

Nonholonomic mechanical systems ,i.e. systems with non integrable differential constraints on the generalized constraints, are a growing field of research, since such applications as space robot arms could show increased ligthness by reducing the number of actuators at joints. Even if nonholonomy is a mechanical property of the system, it has definite effects on the control problem, in this case the configuration space dimension exceeds that of the control space, because of the free joint not equipped with an actuator. The additional property of flexibility, contributing to reduce weight, cost and energy consumption while still maintaining an adequate degree of dexterity, also adds extra difficulties to the control problem by increasing the generalized coordinates and turning the equations of motion more complex.

Two-flexible-link model

The manipulator consists of two links, with the second joint free. Hence, the control is performed only by the torque applied to the first joint .The system's general configuration is shown in Fig. 1. The modeling of the manipulator with flexible links is derived by using Lagrange's equation.

Referring to the figure, the position of any point on

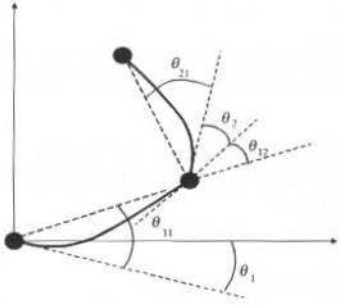


Fig. 1 The 2-link configuration

The first link is given by

$$x_1 = l_1 \cos(\theta_1 + \theta_{11}) \tag{1}$$

$$y_1 = l_1 \sin(\theta_1 + \theta_{11}) \tag{2}$$

The position of any point on the second link is described by the following relation :

$$x_2 = x_1 + l_2 \cos(\theta_1 + \theta_{11} + \theta_{12} + \theta_2 + \theta_{21}) \tag{3}$$

$$y_2 = y_1 + l_2 \sin(\theta_1 + \theta_{11} + \theta_{12} + \theta_2 + \theta_{21}) \tag{4}$$

The parameters of the two-link flexible arm are shown in Table 1.

Table 1 Parameters of the 2-link system

l_1	length of arm 1	0.5 [m]
l_2	length of arm 2	0.5 [m]
m_1	mass of elbow	4.0 [kg]
m_2	mass of payload	4.0 [kg]
J_1	inertia moment of arm1	0.0008 [kg.m^2]
J_2	inertia moment of arm 2	0.00023 [kg.m^2]
J_{11}	inertia moment of elbow	0.0088 [kg.m^2]
J_{21}	inertia moment of payload	0.023 [kg.m^2]
J_{m1}	inertia moment of motor 1	0.068 [kg.m^2]
J_{m2}	inertia moment of motor 2	0.013 [kg.m^2]
E_1I_1	flexural rigidity of arm 1	3.28 [N/m^2]
E_2I_2	flexural rigidity of arm 2	3.28 [N/m^2]
C_1	viscous rotational damping coefficient of arm 1	0.01 [N.s/m]
C_2	viscous rotational damping coefficient of arm 2	0.01 [N.s/m]
C_{11}	internal damping coefficient of arm 1	0.0375 [N.s/m]
C_{21}	internal damping coefficient of arm 2	0.0375 [N.s/m]

To analize the physical model of the system, some assumptions are performed as follow :

a) The deformation of the arm for a simply supported beam with moments at both ends is assumed static.
b) The arm is massless.
c) The shaft friction force is negligibly small to drive the arm.
d) The deformation of the arm is much smaller than the length of the arm.
e) The influence of gravity is ignored.
f) The control stick and the driving stick are rigid bodies and the difference of the angles between them is very small.
g) The motion is restrained to the horizontal plane.
h) The potential energy of the system can be expressed in the form

$$U = \frac{3E_1I_1}{2l_1}\theta^2{}_{12} + \frac{3E_2I_2}{2l_2}\theta^2{}_{22}, \tag{5}$$

and the kinetic energy of the system is as follows

$$\begin{aligned} T = &\frac{m_1}{2}(\dot{x}_1^2 + \dot{y}_1^2) + \frac{m_2}{2}(\dot{x}_2^2 + \dot{y}_2^2) + \frac{1}{2}J_1\dot{\theta}_1^2 \\ &+ \frac{1}{2}J_{m1}\dot{\theta}_1^2 + \frac{1}{2}J_{11}(\dot{\varphi}_1 + \dot{\theta}_{12})^2 + \frac{1}{2}J_2(\dot{\varphi}_2 - \dot{\theta}_{21})^2 \\ &+ \frac{1}{2}Jm_2(\dot{\varphi}_2 - \dot{\theta}_{21})^2 + \frac{1}{2}J_{21}(\dot{\varphi}_2 + \dot{\theta}_{22})^2, \end{aligned} \tag{6}$$

where

$$\varphi_1 = \theta_1 + \theta_{11} \tag{7}$$

$$\varphi_2 = \varphi_1 + \theta_{12} + \theta_2 + \theta_{21} \tag{8}$$

The dissipation of energy can be written in the form :

$$F = \frac{1}{2}C_1\dot{\theta}_1{}^2 + \frac{1}{2}C_2\dot{\theta}_2{}^2 + \frac{1}{2}C_{11}l_1{}^2\dot{\theta}_{11}{}^2 + \frac{1}{2}C_{21}l_2{}^2\dot{\theta}_{21}{}^2 \tag{9}$$

Applying the Lagrangian , L , and Lagrange's equation :

$$T = L - U, \tag{10}$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} + \frac{\partial F}{\partial \dot{q}_i} = \tau, \tag{11}$$

where the set of identified generalized coordinates is

$$q_i = (\theta_1, \theta_2, \theta_{11}, \theta_{21}), \tag{12}$$

with θ_1, the rotation angle of the first arm; θ_2, the rotation angle of the second arm; θ_{11}, the deflection angle of the first link on the shoulder; and θ_{21}, the deflection angle of the second link on the elbow. Then, the equations of motion are obtained in the form :

$$M(\theta)\ddot{\theta} + h(\theta,\dot{\theta}) + (\theta) = B\tau, \tag{13}$$

where, θ is the vector of generalized coordinates, M is the matrix of coefficients, τ the applied torques, and the vectors C, h, and B are as indicated below.

$$q = [\theta_1, \theta_2, \theta_{11}, \theta_{21}]^T, \tag{14}$$

$$M(\theta) = \begin{bmatrix} M_{11} & M_{12} & M_{13} & M_{14} \\ M_{21} & M_{22} & M_{23} & M_{24} \\ M_{31} & M_{32} & M_{33} & M_{34} \\ M_{41} & M_{42} & M_{43} & M_{44} \end{bmatrix} \tag{15}$$

$$h(\theta,\dot{\theta}) = [h_1 \quad h_2 \quad h_3 \quad h_4]^T, \tag{16}$$

$$C(\theta) = [0 \quad 0 \quad \frac{3E_1I_1}{I_1}\theta_{11} \quad 0]^T, \tag{17}$$

$$B = [1 \quad 0 \quad 0 \quad 0]^T. \tag{18}$$

From here, the State Equation form (19) is obtained as follows

$$\frac{d}{dt}\begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_{11} \\ \theta_{21} \\ \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_{11} \\ \dot{\theta}_{21} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ N_1(M_{ij}) \\ N_2(M_{ij}) \\ N_3(M_{ij}) \end{bmatrix} u + \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_{11} \\ \dot{\theta}_{21} \\ 0 \\ P_1(M_{ij}) \\ P_2(M_{ij}) \\ P_3(M_{ij}) \end{bmatrix}, \tag{19}$$

where, "u" is the input, i.e. the torque applied to the first joint, expressed by the acceleration of the angle θ_1.

Conditions of Nonholonomic Nature

The holonomic or nonholonomic nature of the system has to be determined, and for this purpose, the conditions of integrability of the dynamic equation (20) relative to the free joint is examined. Since no input term explicitly appears in equation (20), this may be interpreted as a constraint involving generalized coordinates as well as their first and second-order time derivatives.

$$M_{21}\ddot{\theta}_1 + M_{22}\ddot{\theta}_2 + M_{23}\ddot{\theta}_{11} + M_{24}\ddot{\theta}_{21} + h_2 = 0. \tag{20}$$

For this equation, the property of partial integrability has to be examined. If partial integrability holds, possible further integrability to a constraint of the form f(q,t)=0 must be investigated. If such a constraint exists, the equation (20) is said to possess the complete integrability property, or in other words, to be holonomic, (Nakamura et al.). The ability to discern between holonomic and nonholonomic constraints is crucial, since the former can be used to reduce the system dimension by eliminating some coordinates. In this case, the condition for partial integrability is not fulfilled , since

$$\frac{\partial}{\partial \theta_2}(\dot{\theta}^T M(\theta)\dot{\theta}) \neq 0 \tag{21}$$

Therefore, the system has proved to possess nonholonomic nature, i.e. to be a nonholonomic system.

The Fourier Basis Algorithm

In order to obtain a near optimal solution for the problem of positioning the arm system from an initial configuration to a final configuration, the Fourier basis algorithm is applied. This kind of algorithm has already been applied successfully for computing optimal solutions for a variety of systems. It is assumed that the system is controllable so that the problem is solvable. The nonlinear system can be described by a general expression of the form,

$$\dot{x}(t) = f(x(t), u(t)), \tag{22}$$

with $x(0)=x_0$, and the cost function to be minimized corresponding this system should be written in the form :

$$J = \Delta x_{t_f}^T M \Delta x_{tf} + \int_0^{f\,t} \Delta x^T Q \Delta x dt , \tag{23}$$

with

$$\Delta x_{t_f} = x(t_f) - x_{des}, \tag{24}$$

and

$$\Delta x = x(t) - x_{m,des}, \tag{25}$$

where, $x(t_f)$ is the actual final position , x_{des} is the final desired position and M , Q are the weighting matrices. Rewriting the cost function in terms of the control input u , we have

$$J = (x_f - x_d)^T M (x_f - x_d) + \int_{t_0}^{t_f} u^T R u dt , \tag{26}$$

where the weighting matrices M and R are adequate diagonal matrices. Since the system is controllable, there exists a solution $u \in L_2([0,T])$, here L_2 denotes the Hilbert space of measurable vector –valued functions of the form :

$$u(t) = (u_1(t), \ldots u_m(t))^T , \quad t \in [0,T] . \tag{27}$$

If e_i is an orthonormal basis, e.g., the Fourier basis, a function u , in basis terms, can be expressed as

$$u = \sum_{i=1}^{\infty} \alpha_i e_i , \tag{28}$$

for some sequence $\alpha = (\alpha_1, \alpha_2, \ldots)$. The idea of the Fourier basis algorithm is to approximate the solution by solutions of some finite-dimensional systems, introduced by restricting the control to the first n terms of the basis, (Fernandes et al., 1992). By applying the Fourier basis method, we obtain the following expressions :

$$u(t) = \sum_{i=0}^{n} \alpha_i e_i \tag{29}$$

$$u(t) = \alpha_0 + \alpha_1 + \alpha_2 + \cdots + \alpha_n + \cdots \tag{30}$$

$$u(t) = \frac{a_0}{2} + \sum_{i=1}^{n} (a_i \cos nt + b_i \sin nt). \tag{31}$$

It is demonstrated that as $n \to \infty$, solutions of the finite dimensional systems converge to the optimal solution. Finally, the cost function can be expressed in a general form as

$$J(u(t)) = J(a_0, a_1, a_2, \cdots\cdots, a_n, \cdots). \tag{32}$$

This variational approach to obtain a direct way to solve the problem is the Ritz method.

Now, the algorithm to solve for optimal α, that is, $\alpha \in l_2$ of minimum cost linking initial and final configurations will be constructed. By making R=1, with a control time $t=2\pi$, and considering the Fourier basis in

$$\int_{t_0}^{t_f} e_i(t)^T e_j(t)\,dt = \begin{cases} 0 (i \neq j) \\ 1 (i = j) \end{cases}. \tag{33}$$

Replacing the expression for u in equation (26), the rewritten expression is :

(34)

At this point, an efficient approach to minimizing $J(\alpha)$ is quadratic programming. For this effect, the computation of the Hessian about a point α_n is developed using its Taylor expansion :

$$J[\alpha_n + \delta] = J[\alpha_n] + \left\langle \left.\frac{\partial J}{\partial \alpha}\right|_{\alpha_n}, \delta \right\rangle + \frac{1}{2}\left\langle \left.\frac{\partial^2 J}{\partial \alpha^2}\right|_{\alpha_n} \delta, \delta \right\rangle + 0(\|\delta\|^3), \tag{35}$$

where

$$\left.\frac{\partial J}{\partial \alpha}\right|_{a_n} = 2(\alpha_n + Y_f^T M(x(t_f) - x_d)), \tag{36}$$

and

$$\left.\frac{\partial^2 J}{\partial \alpha^2}\right|_{a_n} = 2(I + Y_f^T M Y_f + \sum_{i=1}^{n} Z_{if}(M(x(t_f) - x_d))_i). \tag{37}$$

Then, the Jacobian Y(t) and the Hessian $Z_i(t)$ can be expressed by eqs. (38) and (39) respectively

$$Y_f = Y(t) = \frac{\partial x(t)}{\partial \alpha} \tag{38}$$

$$Z_{if} = Z_i(t) = \frac{\partial^2 x_i(t)}{\partial \alpha^2}. \tag{39}$$

Since an expression for the differential equation for Y is needed, it can be obtained from the expressions (29), (38), and the general form of the state equation :

$$\dot{x} = h(x)u + g(x) = h(x)E(t)\alpha + g(x), \tag{40}$$

with $E = (e_1(t), e_2(t), \ldots, e_n(t))$; then

$$\begin{aligned}
\frac{\partial Y}{\partial t} = \dot{Y} &= \frac{\partial}{\partial t}\frac{\partial x}{\partial \alpha} = \frac{\partial}{\partial \alpha}\frac{\partial x}{\partial t} = \frac{\partial}{\partial \alpha}\dot{x} = \frac{\partial}{\partial \alpha}(h(x)u + g(x)) \\
&= \frac{\partial h(x)}{\partial \alpha}u + h(x)\frac{\partial u}{\partial \alpha} + \frac{\partial g(x)}{\partial \alpha} = \sum_{i=1}^{r}\left(\frac{\partial h(x)}{\partial \alpha}u_i\right) + h(x)\frac{\partial (E\alpha)}{\partial \alpha} + \frac{\partial g(x)}{\partial \alpha} \\
&= \sum_{i=1}^{r}\left(\frac{\partial h(x)}{\partial x}\frac{\partial x}{\partial \alpha}u_i\right) + h(x)E + \frac{\partial g(x)}{\partial x}\frac{\partial x}{\partial \alpha} = \sum_{i=1}^{r}\left(\frac{\partial h(x)}{\partial x}Yu_i\right) + h(x)E + \frac{\partial g(x)}{\partial x}y \\
&= \left(\sum_{i=1}^{r}\left(\frac{\partial h(x)}{\partial x}u_i\right) + \frac{\partial g(x)}{\partial x}\right)Y + h(x)E
\end{aligned} \tag{41}$$

Finally, in order to obtain an expression to update α, the modified Newton's method is used

$$\alpha_{n+1} = \alpha_n - \mu\left(\frac{\partial J}{\partial \alpha}\Big|_{\alpha_n} \Big/ \frac{\partial^2 J}{\partial \alpha^2}\Big|_{\alpha_n}\right). \tag{42}$$

Since the Hessians Z_{if} of the component functions are difficult to compute, by applying Newton these terms can be ignored. Then the expression for the updated α becomes

$$\alpha_{n+1} = \alpha_n - \mu\frac{\left[\alpha_n + Y_f^T M(x(t_f) - x_d)\right]}{\left[I + Y_f^T M Y_f\right]}, \tag{43}$$

where $\mu \in (0,1)$ is a parameter.

Summarizing the steps to follow to construct the basis algorithm :

a) Input : The initial value x_0 , the final desired value x_d, and the system's equation.
b) Output : The control input linking x_0 and x_d ; in this case, the Fourier parameter α .
c) Choose an orthonormal basis , in this case the Fourier basis, and retain the first n elements, to set the order of the series.
d) Choose the parameter of the Newton's method μ ,and solve the equations of the system as well as the calculation of the Jacobian. Examine the obtained data.
e) If the last $x(\alpha_n)$ satisfies the expected objective x_d, exit; otherwise repeat from step (d).

Positioning the Flexible Arm

The proposed method to attain an optimal control of the motion of the 2-flexible-link nonholonomic manipulator is applied for four cases. The initial and final configurations are listed in Table 2. The state variable vector is

$$x = [\theta_1, \theta_2, \theta_{11}, \theta_{21}, \dot{\theta}_1, \dot{\theta}_2, \dot{\theta}_{11}, \dot{\theta}_{21}]^T. \tag{44}$$

Table 2 Initial and Final Position

	Initial Position x_0	Final Position x_d
I	(0,0,0,0,0,0,0,0)	(20,0,0,0,0,0,0,0)
II	(0,0,0,0,0,0,0,0)	(20,20,0,0,0,0,0,0)
III	(25,0,0,0,0,0,0,0)	(50,20,0,0,0,0,0,0)
IV	(30,0,0,0,0,0,0,0)	(75,0,0,0,0,0,0,0)

Then the Fourier basis became :

$$E = \{1/2, Sin[1t], Cos[1t], Sin[2t], Cos[2t], \cdots, Sin[30t], Cos[30t]\} . \quad (45)$$

The weighting matrices used for each case are shown in the Table 3. The time parameters are : t_0=0, and $t_f = 2\pi$.

Table 3 Weighting matrices

	Weighting Matrix
I	M=diag[3000,3000,3000,3000,0,3000,3000,3000]
II	M=diag[3000,3000,3000,3000,0,3000,3000,3000]
III	M=diag[2000,2000,2000,500,0,500,500,500]
IV	M=diag[2000,500,500,500,0,2000,2000200]

Simulation Results

The results of the simulation are shown in the time history plots for the angles θ_1, θ_2 ,θ_{11}, θ_{21} and the corresponding velocities. Also the plots for the Time History of the input control torque are shown as well as a view of the animation for each case. Firstable, the plots for the Time History of angles are shown in Figs. 2 to 5. For all cases, the dashed line represents the angle θ_1; the dashed-and-dotted line represents the angle θ_2 ; the dotted line is deflection angle θ_{11} ; and the solid line is the deflection angle θ_{21}.

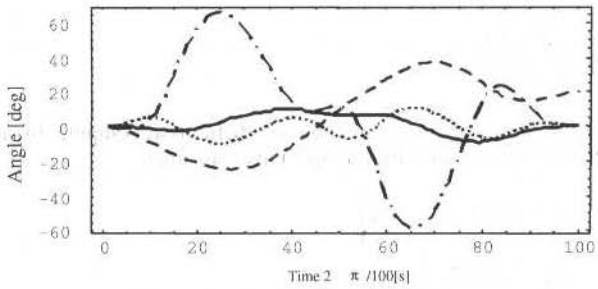


Fig. 2 Case I, Time history of angles

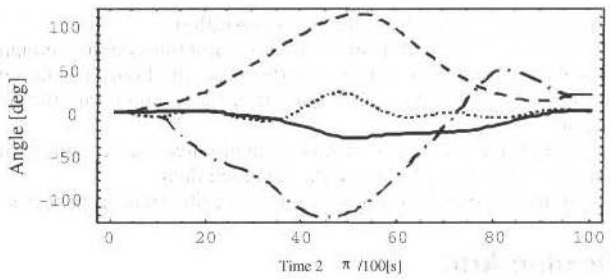


Fig. 3 Case II, Time history of angles

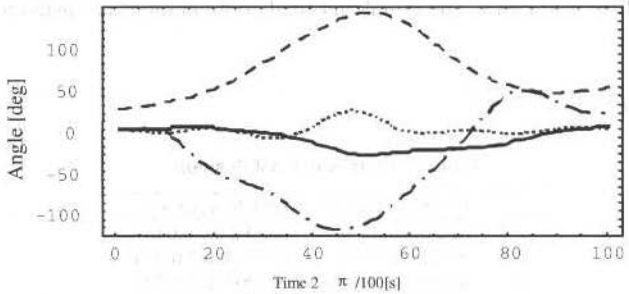


Fig. 4 Case IIII, Time history of angles

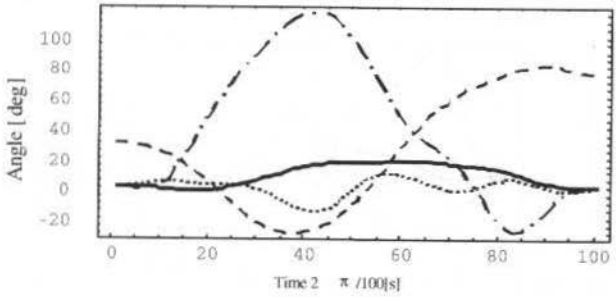


Fig. 5 Case IV, Time history of angles

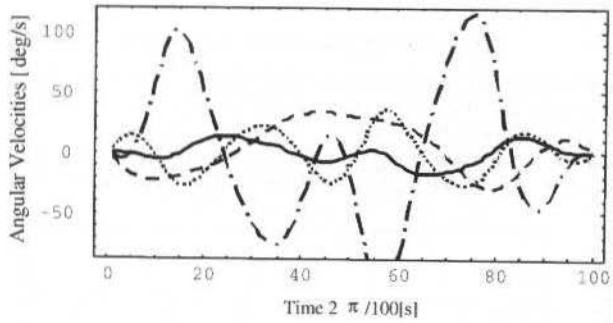


Fig. 6 Case I, Time history of velocities

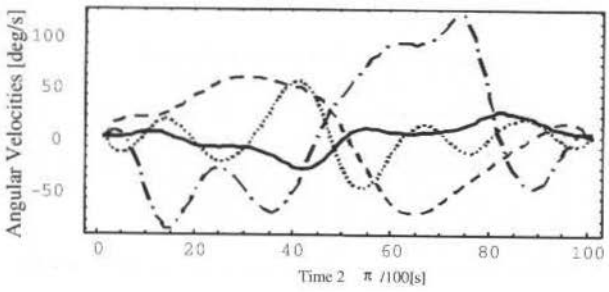


Fig. 7 Case II, Time history of velocities

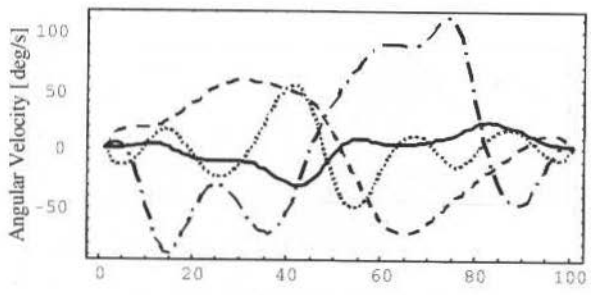


Fig. 8 Case III, Time history of velocities

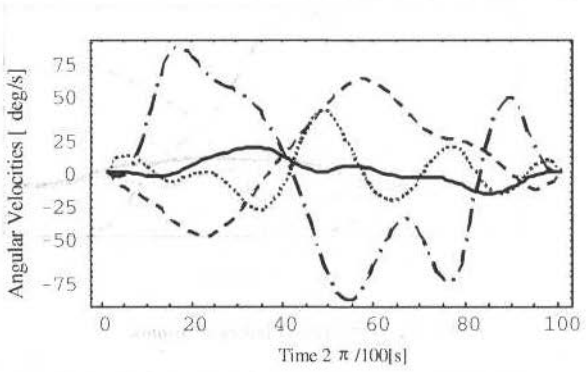


Fig. 9 Case IV, Time history of velocities

Next, the plots for the velocities of the angles are shown from Fig. 6 to 9. As for the angles' plots, the dashed line represents the velocity of θ_1, the dashed-dotted the velocity of θ_2, the dotted line the velocity of θ_{11} and the solid line the velocity of θ_{21} .

The input control, the torque applied to the first joint, obtained through the algorithm is represented in Figs. 10 to 13, for each case. The abscissa indicates the division of time and the ordinate indicates the torque.

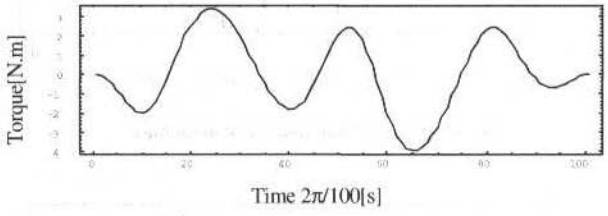


Fig. 10 Case I, Time history of torque

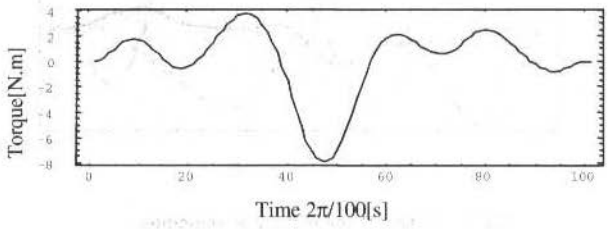


Fig. 11 Case II, Time history of torque

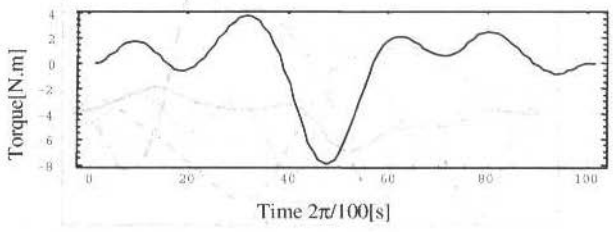


Fig. 12 Case III, Time history of torque

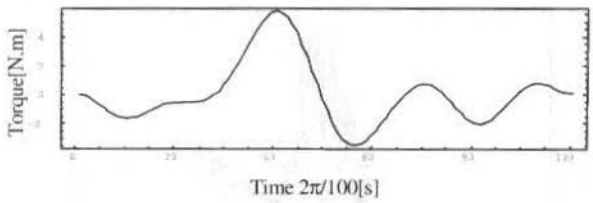


Fig. 13 Case IV, Time history of torque

From the figures,we can see a similar pattern for cases II and III; the applied torque and the angular velocities have very close values. This could indicate that within certain range the behavior is almost the same despite the different targets. Except the first case, the curves for torque show only one peak value

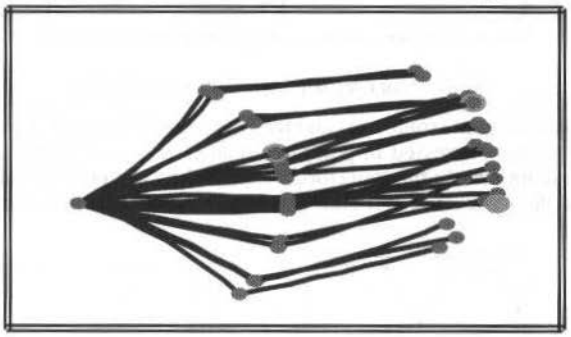


Fig.14 (0,0,0,0,0,0,0,0) →(20,0,0,0,0,0,0,0)

about the point where each link reaches its maximum angle. Finally, the animation of the controlled motion from the initial position to the desired target is shown in the following views, Figs. 14 to 17.

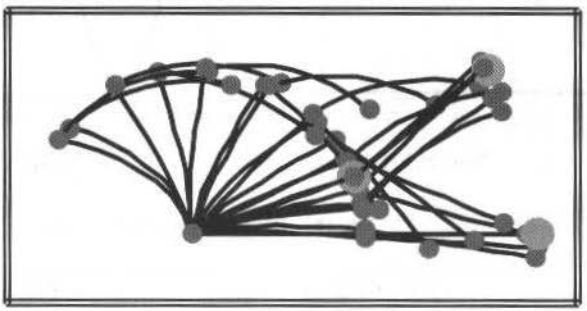


Fig.15 (0,0,0,0,0,0,0,0) → (20,20,0,0,0,0,0,0)

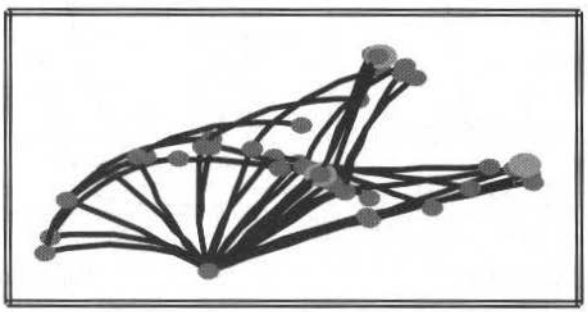


Fig. 16 (25,0,0,0,0,0,0,0) → (50,20,0,0,0,0,0,0)

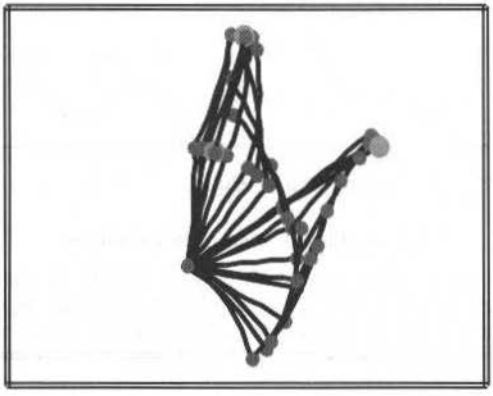


Fig. 17 (30,0,0,0,0,0,0,0) → (75,0,0,0,0,0,0,0)

The position of the second joint (solid line) and the position of the tip of the second link (dotted line) obtained in each case, are also expressed in polar coordiantes (r,θ) and shown in Figs. 18 to 21. From these graphs we can see more clearly the trajectories followed by the points, and notice the similarity of patterns among some of the cases. The left side graph represents the radius r and the right side graph represents the angle θ.

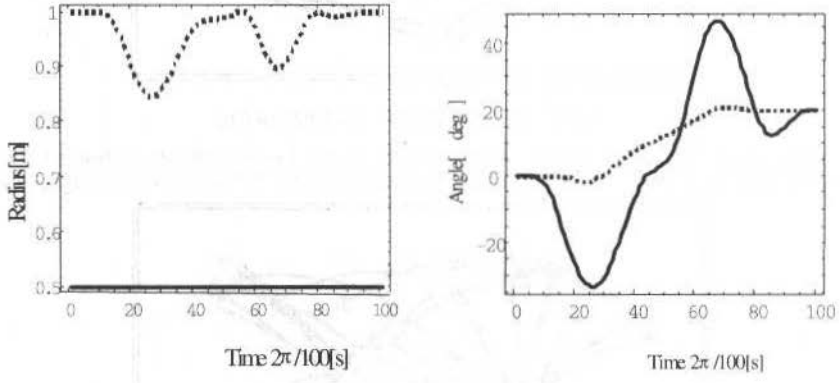


Fig. 18 Case I

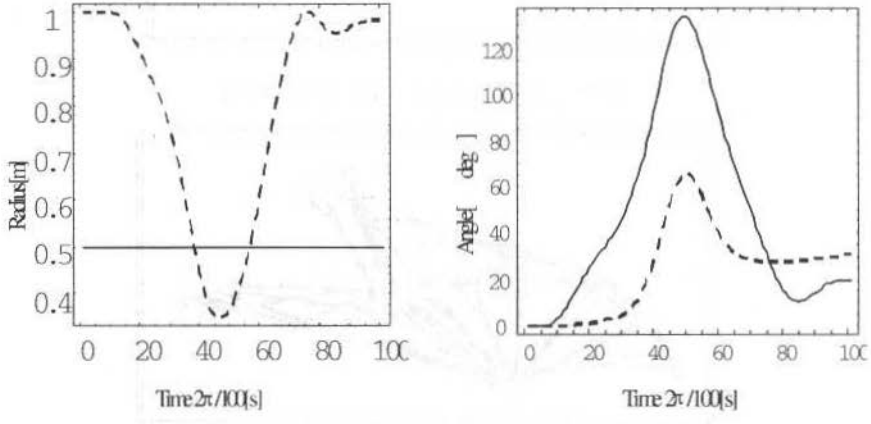


Fig. 19 Case II

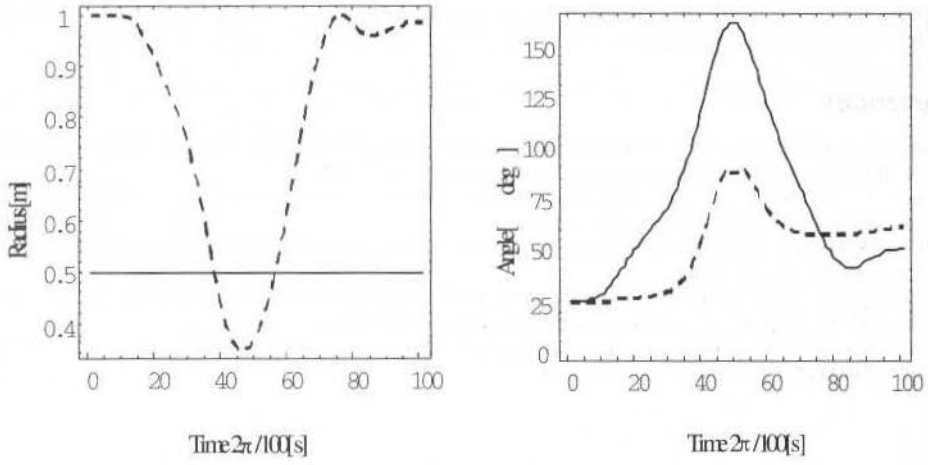


Fig. 20 Case III

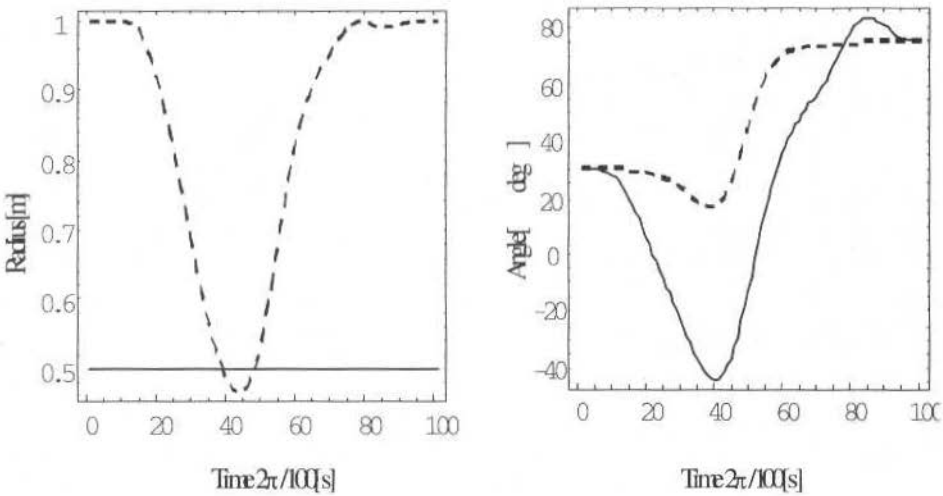


Fig. 21 Case IV

Conclusions

The proposed Fourier basis algorithm has been applied to solve the problem of finding the near optimal control input necessary for the accurate positioning of a two-flexible-link manipulator with a free joint. The nonholonomic nature of the system is one of the conditions to take into account in order to attain a reliable response. The modeling of the manipulator was derived using Lagrange principle, in addition to this, the flexibility of the links has been introduced considering the deformation of a beam. Besides the construction of the algorithm itself, where the parameters μ and the choice of the order of the Fourier series are important steps to reach the closest possible solution to the optimal solution, the choice of the weighting matrix M is also an important step in order to acquire significant and valid results. In this case, the results obtained through this method showed the validity of the procedure, since they are attained taking into account the conditions of nonholonomy and flexibility, two characteristis that are usually studied separately. Also by minimizing the cost function involving the energy and the position error, the method achieves a near optimal solution for the problem of controlling the motion and position of this kind of underactuated mechanism. The possibility of applying the algorithm for a multiple-link

manipulator, in other words, to generalize the method for more than two links is being considered due to the effectiveness of its achievements.

References

Fernandes C. and Gurvits L. 1992, "Attitude Control of Space Platform/Manipulator System Using Internal Motion". Proc. IEEE Int. Conference on Robotics and Automation, pp. 893 – 898 .

Nakamura Y. and Oriolo G. "Free-Joint Manipulators : Motion Control Under Second Order Nonholonomic Constraints". Proc. IRO'S 91, pp. 1248 – 1253.

Cao, H. and Yoshida K., 1996, "The Trajectory Tracking Control of 2-Link Flexible Arm Using Tendom Mechanism". ISCA, Vol. 3, No. 1, pp. 1-9.

Fernandes C. , Gurvits L. and Li Z. X., 1991 "A Variational Approach to Optimal Nonholonomic Motion Planning". Proc. IEEE Int. Conference on Robotics and Automation; pp. 680 – 686.

Brockett R. W., 1983 "Asymptotic Stability and Feedback Stabilization in Differential Geometric Control Theory". Progress in Mathematics, No. 27 pp. 181 – 191 .

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 485-494

ISSN 0100-7386
Printed in Brazil

The Use of Genetic Algorithms on a Fuzzy Controller for a Satellite Attitude Control During the Pointing Phase

Cláudio Correa
Sandra Aparecida Sandri
Luiz Carlos Gadelha de Souza
Applied Computing CAP -- Space Mechanics & Controls Division DMC
Brazilian National Institute for Space Research – INPE
CP 515 -- São José dos Campos, 12227-010, Brazil
correa@pgrad.inpe.br, gadelha@dem.inpe.br, sandri@lac.inpe.br

Abstract

In this paper, we propose the use of genetic algorithms (G.A.) for the creation of controllers for the pointing phase of a reaction wheel artificial satellite. We make use of the simulator previously developed which test a PD controller for the pointing phase of a satellite similar to the French-Brazilian Satellite, whose control system is based on the stabilization of 3 axes.
Keywords: *Genetic Algorithm, Fuzzy Control, Satellite.*

Introduction

There exists nowadays a technological tendency of building small artificial satellites, aiming at guaranteeing a fast and simple means of reaching space. The tendency is to have these satellites equipped with highly autonomous systems for attitude, maneuver and orbit control, and to have them developed fastly and at very low costs (Guerra, et al., 1999).

In this work, we make use of a model similar to the French-Brazilian (Souza, et al., 1999) satellite one, currently under development at the Brazilian National Institute for Space Research (INPE). This satellite has its control system based on the stabilization of three axes and makes use of a proportional/derivative controller (PD), whose gains have been determined through the poles allocation method, and whose sensor measures have been processed by a Kalman filter (Silva, 1997).

It is important to observe that in contrast to the high complexity of a controller system, it is imperative for it to have the lowest possible cost. For this reason, it is important to investigate different techniques in the development and implementation of a controller. Such innovations can bring versatility in what regards the hardware components to be employed, and in relation to the interfaces among controllers, sensors and actuators.

Fuzzy logic is a one of the most well-succeeded recent technologies in the development of sophisticated control systems (Driankov, 1993) (Lee, 1990). Its employment, complex requirements can be implemented in simpler controllers, of easy maintenance and low cost.

The augmentation of satellite autonomy has been long pursued with the goal of not only improvement in performance, but especially in order to reduce fuel consumption. To obtain this autonomy, one approach is the use of fuzzy controllers, especially when the model is subject to uncertainty (Woodward, July 1996) (Conway, et al., 1994) (Guerra, et al., 1997) (Guerra, et al.).

A fuzzy controller is composed of a set of rules of the type If *<premise>* then *<conclusion>*, which define control actions in function of some (usually ill-defined) intervals on which the state variables may take their values. These intervals are modeled by fuzzy sets and called fuzzy terms.

The main difficulty in the creation of fuzzy controllers is the definition of the fuzzy terms. One way of dealing with this problem is to use *"neuro-fuzzy"* models (Jang, 1993) (Lin, 1995), in which these parameters are learned through the presentation of pairs (input, expected output) to a neural network with nodes that basically compute the intersection and union operations. Another way to learn these parameters is to employ genetic algorithms.

In this paper we present two attitude controllers for the French-Brazilian satellite, built with the use of genetic algorithms. The first one is a PD controller whose gains were found using a genetic algorithm. The second one is a fuzzy controller of the Mamdani type, whose parameters have been learned with another genetic algorithm, using however the same fitness function of the first one.

In the present work, we have used the satellite simulated model, which has been previously developed at INPE using the MATLAB toolkit (Silva, 1997). We have also taken the PD attitude controller originally developed with the simulator as basis of comparison to assess the quality of the results obtained by the GA based controllers presented here.

This paper is divided as follows. In Section II we present some fundaments about genetic algorithms and fuzzy controllers. In section III we present the model of the satellite used in our applications, and the original PD attitude controller developed for it in the pointing phase. In Section IV we present the PD and the fuzzy controllers developed using genetic algorithms for the satellite model. Finally, Section V brings the conclusions.

Basic Notions

Genetic Algorithms

Genetic algorithms are adaptive search strategies based on a highly abstract model of biological evolution (Husbands, 1992). They are primarily used in optimization problems for which one aims to find not necessarily an optimal solution, but at least a reasonably good solution.

In these algorithms, a population of individuals (potential solutions) suffers a series of unary transformations (mutation) and of higher order (crossover). These individuals compete among themselves for survival; the most apt individuals have better chances to be chosen to pass their characteristics to the next generation. After some generations, the algorithm (usually) converges and the best individual represents a solution close to the optimum.

The search for the solution involves an evaluation function (fitness), which yields a grade for the performance of each individual, according to aspects considered relevant to the problem at hand. Figure 1 brings an illustration (Vasconcelos, et al., 1997) of an evolution cycle in a genetic algorithm.

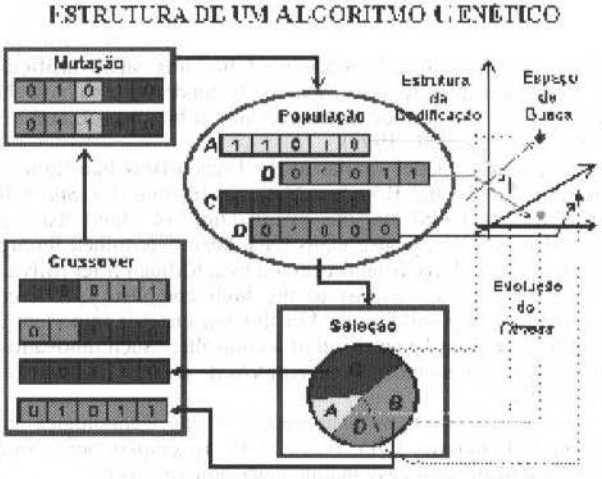


Fig. 1 Structure of a simple genetic algorithm

Genetic algorithms have been used in many applications involving fuzzy control (Karr, 1991) (Karr, 1994) (Pham, et al., 1991) (Kinzel, et al., 1994) (Kim, et al., 1994) (Mohamadian, et al., 1994) (Mohamadian, et al., 1994), inclusive in Brazil (Aya, et al., 1997) (Silva, et al., 1997) (Vasconcelos, et al., 1997). In the present work, the fitness function of the G.A. for the pointing phase is a global measure of the performance of each solution in relation to the simulation of a complete orbit under a certain of perturbation.

Fuzzy Controllers

Fuzzy controllers are based on fuzzy sets theory, which has been developed since 1965 after the seminal works of Lotfi Zadeh (Zadeh, 1965). Fuzzy control techniques were first developed with the works of E.H. Mamdani (Mamdani, 1976) (Mamdani, et al., 1975) (Mamdani, et al.), and have been gaining increasing importance over the years, being today the main application of fuzzy sets theory. The term fuzzy logic is usually employed in the control field to name the modeling of fuzzy pieces of information and the inference mechanisms that act upon them.

Contrary to what happens in conventional control in which the control algorithm is described analytically by algebraic or differential equations, by means of a mathematical model, in fuzzy control logical rules are employed in the control algorithm, obtained through the synthetization of human experience, intuition and heuristics, in process control (Zadeh, 1965).

A rule in a fuzzy controller is usually of the type *If* $x_1 = A_1$ *and* $x_2 = A_2$ ... *and* $x_n = A_n$ *then* $y = B$, where the x_i and y are respectively state and control linguistic variables, and the A_i's and B are linguistic terms. A linguistic variable is a 4-tuple $(x, T(x), \Omega, M)$, where x is the name of the variable, Ω is the domain of x, $T(x)$ is a set of linguistic terms, i.e. a set of names of fuzzy sets, and M is a function that associates a fuzzy set in Ω to each term in $T(x)$. Figure 2 brings illustrates the linguistic variable "*error*" with terms $T(error)$={*negative_big, negative_small, zero, positive_small, positive_big*}. A rule in a fuzzy controller could be for instance be "*If error = small then throttle = big*".

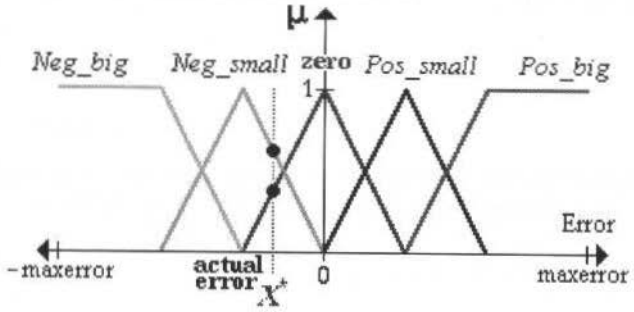


Fig. 2 Linguistic terms of variable "throttle"

The basic structure of a fuzzy controller is illustrated (Lee, 1990) in Fig. 3. The main components are the knowledge base that contains the rules and the description of the linguistic variables, and the inference engine, that allows us to obtain a control action in function of the value of the state variables in a given moment of time.

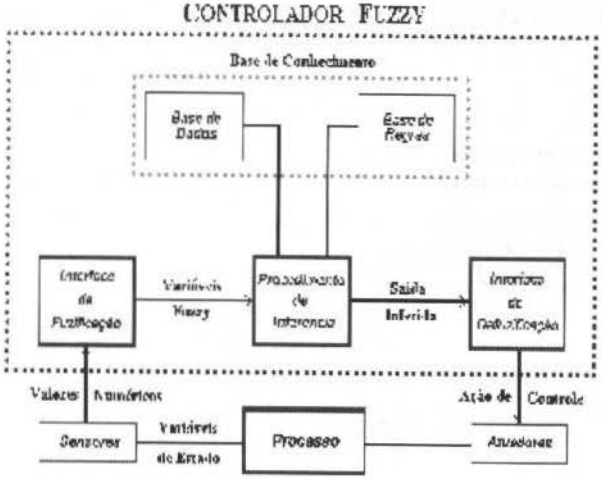


Fig. 3 Structure of a fuzzy controller

Fuzzy controllers are highly adaptable and capable of incorporating knowledge that many other systems are incapable of doing (Guerra, 1998). They are also versatile, especially when the physical model is very complex and of difficult mathematical reproduction.

In general, they are more useful in non-linear systems, varying in time or not, support very well perturbations and highly noisy plants, and are robust even in systems where uncertainty is intrinsically present.

Satellite Model and Original PD Controller

Satellite Model

The French-Brazilian satellite model is a rigid body model where the null inertia product is neglected. The control system consisted of a gyro and a star and sun sensors plus a PD controller in the pointing

phase. The rotational dynamics of the satellite can be represented by the following differential equations system (Silva, 1997):

$$\begin{aligned} \dot{\xi} &= \varpi \\ \dot{\varpi} &= -K_P * \hat{\xi} - K_D * \hat{\varpi} + T_P(t) \\ \dot{\hat{\xi}} &= \hat{\varpi} \end{aligned} \tag{1}$$

where $\hat{\xi}$, $\hat{\varpi}$ and $T_P(t)$ represent the respective estimation of rotation (ξ), angular speed (ϖ) and perturbation torque's. The estimation of the angular speed (ϖ) is determined with:

$$\hat{\varpi} = o - \hat{b}_k \tag{2}$$

and the gyro output (o) is given by:

$$o(t) = \varpi + b_k + n_k; \qquad t \in [t_k; t_{k+1}) \tag{3}$$

where υ_k represents the long period bias (derive) and n_k the short period bias (noise).

From Eq. 2 and Eq. 3 we have:

$$\hat{\varpi} = \varpi + b_k - \hat{b}_k + \sigma_n^2 \tag{4}$$

where σ is the covariance of noise n.

The long period bias and its estimation can be propagated in time with:

$$\begin{aligned} b_{k+1} &= e^{-\lambda * \Delta t} * b_k + n(2 * \lambda * \Delta t * \sigma_d^2 \\ \hat{b}_{k+1} &= e^{-\lambda * \Delta t} * \hat{b}_k \end{aligned} \tag{5}$$

where λ represents the correlation and υ_d the long period covariance.

Making $D_k = b_k - \hat{b}_k + n(\sigma_n^2)$, Eq. 1 becomes:

$$\begin{aligned} \dot{\xi} &= \varpi \\ \dot{\varpi} &= -K_P * \hat{\xi} - K_D * (\varpi + D_k) + T_P(t) \\ \dot{\hat{\xi}} &= (\varpi + D_k) \end{aligned} \tag{6}$$

The coordinate system for the satellite attitude control adopts an external referential, defined as:

- Z_E oriented to the north pole of the ecliptic;
- X_E pointing to the sun position and
- Y_E pointing in such a way as to form a destrogerous system.

The simulations made with a PD controller used the following parameters (Silva, 1997):

1. Pointing precision:
   - 0.5° for axis x;

- 0.15° for axes y and z.

2. Stability:
   - 0.05°/s for the 3 axes.

The following orbital data has been used:

- Altitude at apogee 1500 km;
- Altitude at perigee 400 km;
- Inclination of 7°;
- Argument at perigee of 90°;
- Longitude of the descending node of -90° and
- Mean anomaly of -90°.

Original PD Controller

In the present work, we have used the satellite simulated model, which has been previously developed at INPE using the MATLAB toolkit (Silva, 1997). We have also taken the PD attitude controller originally developed with the simulator as basis of comparison to assess the quality of the results obtained by the GA based controllers presented here.

In the following we present the he original attitude controller for the pointing phase. It consists of a PD controller for each axis, using the following equation:

$$f(t) = K_P * e(t) + K_D * \Delta e(t) \tag{7}$$

where $e(t)$ is the signal error and $\Delta e(t) = d/dt[e(t)]$, K_P and K_D are the proportional and derivative gains.

In order to determine the gains of this controller the Pole allocation method has been employed, which through project restrictions such as peak time (T_P) and accommodation time (T_S), allows us to obtain the position state vector (ξ) and the angular speed (ϖ).

The gains have then been calculated as a function of these specifications, which become implicit in the state vector. Equal gains (proportional and the derivative) have been used for X and Y axis, in order to simplify calculations.

The PD controller responds proportionally to the angular position error $(\zeta(t))$ and its derivative error, which is equivalent to the angular speed $(\Delta\zeta(t))$. The tuning has been performed through the selection of the gains which lead to a satisfactory answer, i.e. the control function ought to maintain the controlled variables as close as possible to the desired values.

The satellite dynamical equations consider the perturbations due to 3 torque's: atmospheric drag, solar radiation pressure and magnetic field. These torque's originate the errors at the satellite pointing, and the function of the controller is to react to the perturbations sending a signal to the atuactors, which then generate an action to correct the pointing along the orbital trajectory.

The gains found in the original model are:

- Proportional gains for the X and Y axes, $K_t = 0.0263$;
- Derivative gains for the X and Y axes, $K_t = 0.08$;
- Proportional gain for the Z, $K_{tz} = 0.0272$;
- Derivative gain for the Z, $K_{wz} = 0.1$.

G.A. Built Attitude Controllers for the Satellite Pointing Phase

In the following we present the G.A. built attitude controllers for the satellite-pointing phase. We first present the simulation conditions used in both developments, to then present the PD and the fuzzy attitude controllers developed using G.A's.

Simulation Conditions

In order to effectively assess the quality of the controllers built with the G.A.'s. severe conditions have been adopted for the satellite operation mode. In this sense, the magnitude order of the perturbations

has been incremented in relation to those adopted to assess the quality of the original PD controller (Silva, 1997). The perturbations due to atmospheric drag and solar radiation pressure have been multiplied by constants $\beta_a = 10$ and $\beta_s = 100$ respectively, corresponding to the value necessary to reach the same order of magnitude of the perturbations due to the magnetic field, which had so far a predominant role over the other values (see Fig. 4).

The satellite state vectors at the initial moment represents a zero error condition, in what regards both pointing and speed. In the present work, an error close to the limits imposed by the project specifications has been adopted for the position of the 3 axes, with speed kept as in the original situation. The initial values for the 3 axes are: $X = 0.35^{\circ}$, $Y = -0.12^{\circ}$ and $Z = 0.12^{\circ}$.

The attitude error relative to the control actions in the simulation of an orbit are shown in Fig. 5 and Fig. 6, calculated as the original control laws (Silva, 1997). To evaluate robustness of the G.A. built controllers, we compare their results with the ones obtained by the original PD controller.

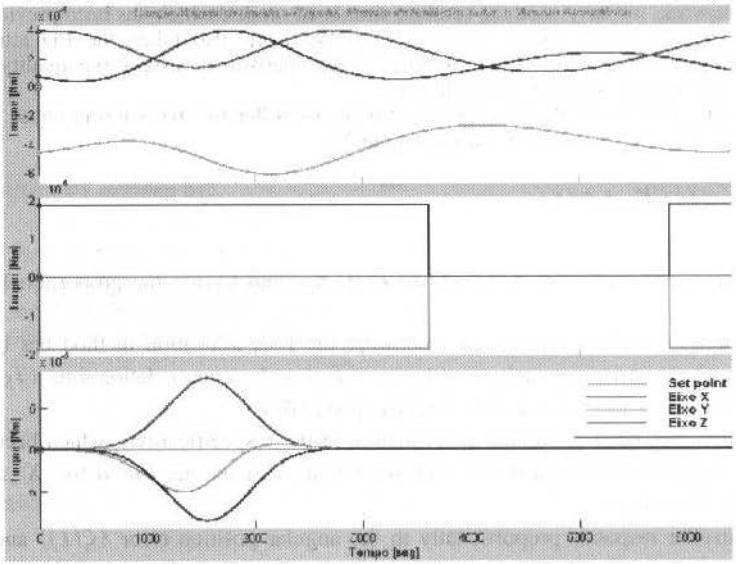


Fig. 4 Perturbation torque's multiplied by β

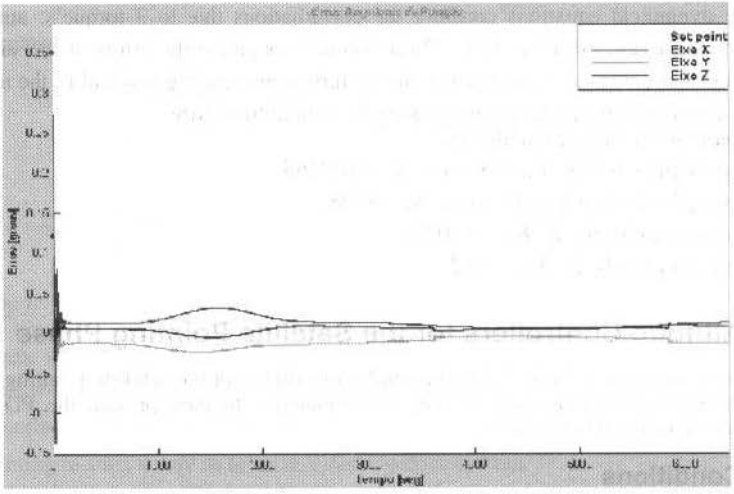


Fig. 5 Reference position errors for the simulations

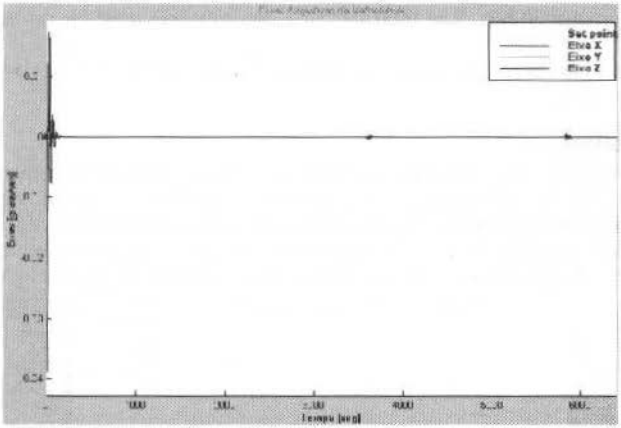


Fig. 6 Reference speed errors for the simulations

G.A. Characteristics

A GAS has been used to built the controllers for the satellite model in this work. Each chromosome in a given population is composed of a set of parameters specifying a controller. In the case of the PD controller, the chromosomes will contain the proportional and derivative gains, and in the case of the fuzzy controller, they will contain the fuzzy terms employed by the rules.

The fitness function of the GAS employed to learn the controller parameters determines a measure on the performance of the candidate controllers, each of which built using the parameters specified in its respective chromosome. This function needs to take into account the relevant aspects in the controller answer, in order to guarantee performance and stability.

Since the fitness function depends only on the results of the simulation of each candidate controller, the same fitness function can be used to optimize different types of controllers. In particular, in this work, a single fitness function has been employed to obtain optimized PD and fuzzy controller.

In order to simplify the evaluation, only variables (ζ,ω) are used in the function. Moreover, all candidate solutions violating a project specification are eliminated.

- Position error restrictions (ζ):
  - Eliminate chromosomes for which $\zeta_X > 0.5^\circ$, or $\zeta_Y > 0.15^\circ$, or $\zeta_Z > 0.15^\circ$;
- Angular speed error restrictions (ω):
  - Eliminate chromosomes for which $\omega_X > 0.05^\circ / s$ or $\omega_Y > 0.05^\circ / s$, or $\omega_Z > 0.05^\circ / s$.

The performance of the GAS was studied using different fitness functions. The best results were obtained using an approximation of the angular position error. This is done by function *trapz.m*, available in Matlab, which calculates a numerical integral by the trapezoidal sum method. An abbreviated version of the fitness function is given by:

$$fitness = \left[1 - \frac{\sum_{i=1}^{n} \left(\int_0^{t_f} f(\theta)d\theta \right)_i * v_i}{0.15 * n} \right] \tag{8}$$

where $f(\theta)$ represents the position error for each of the three, v_i corresponds to weights for each axis, is the number of variables (axes) and t_f is the time taken by the simulated orbits.

In all the tests, the same parameters for the GAS have been maintained: population composed of 30 individuals, $p_c = 0.9$, $p_m = 0.033$, sensibility of 3 decimal cases to detect fitness variation and as last stop criteria a maximum number of 35 generations. Each chromosome is represented by a bit vector, with each

gene (parameter) occupying 20 bits. The total size of a chromosome depends on the number of codified parameters.

Each chromosome has been evaluated in relation to the simulation of 1/4 of an orbit, which correspond to 1600 seconds.

G.A. Built Controllers

We have used the simulation conditions and the GAS structure described above to generate the gains of a PD controller, with a control law for each axis. Therefore, 8 gains were codified in each chromosome. The GAS converged after 23 generations; the results of this controller are shown in Fig. 7.

With the same GAS parameters specification and fitness function given above, we have developed fuzzy controllers of the Mamdani type, using the fuzzy control toolbox available in Matlab.

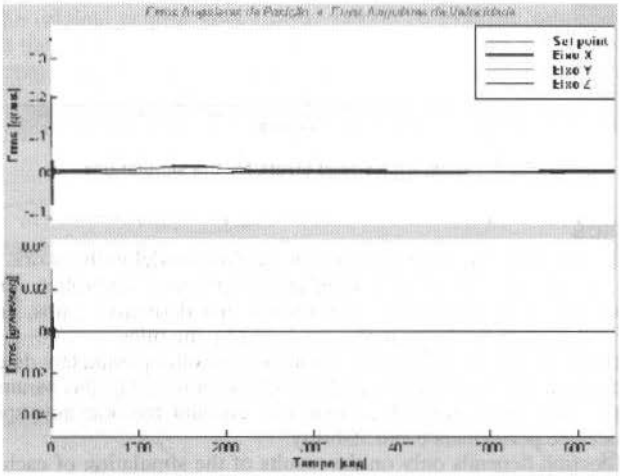


Fig. 7 Errors obtained with the use of a PD controller optimized by a GAS

In this case, the number of fuzzy rules is fixed, as well was the identity of the terms appearing on each rule. The rules were built through the observation of the PD controller. The chromosomes then encode the centers of the fuzzy terms appearing in the premises and conclusion of the rules.

All the fuzzy sets are triangular, except those on the extremities, which are trapezoidal in shape. All fuzzy sets are symmetrical in relation to 0, and therefore, only the positive centers have to be learned. To confer more smoothness to the control surface, neighboring the fuzzy sets are superposed. These restrictions do not limit the model, and have been frequently used in fuzzy controllers (Driankov, 1993) (Heider, et al., 1995) (Larsen, 1998) (Sanchez, et al., 1997).

To further reduce the number of parameters to be learned we have used a single control for all the axes. The surface of best Mamdani fuzzy controller obtained, with 3 fuzzy terms for each input variable and 5 fuzzy terms the output variable (torque), is depicted in Fig. 8. The results of this controller are shown in Fig. 9.

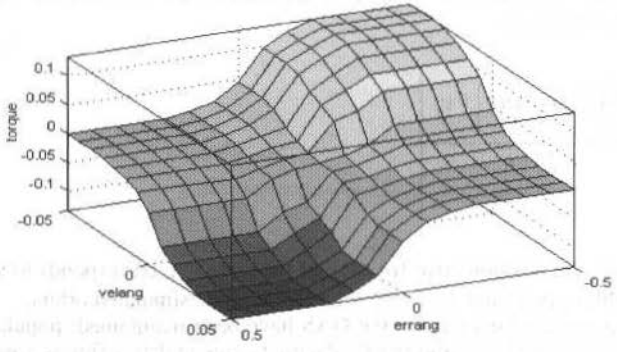


Fig. 8 Mapping of the output torque of the fuzzy controller

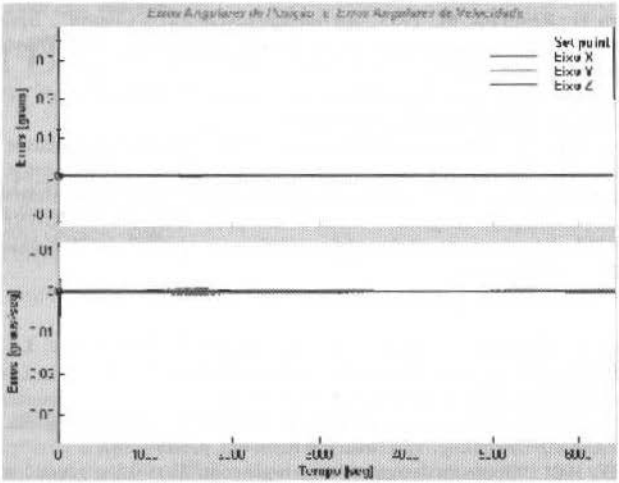


Fig. 9 Errors obtained with the use of a fuzzy controller optimized by a GAS

Conclusions

We have presented a PD and a fuzzy controller for attitude control of a satellite model, whose parameters have been obtained through the use of a simple genetic algorithm. Table 1 below brings a comparison of the performance of these controllers in terms of the sum of the errors yielded by each controller during the simulation of a single orbit. The table also brings the performance of the original PD controller developed for the satellite model using the poles allocation method. It is easy to see that the pointing error is decreasing in PD, PD-GA and CN-GA controller sequence. However, this not happens in the velocity errors as for the CN-GA controller, although, it keeps the velocity errors under specifications design.

Table 1 Results of the integral of the errors and total sum of the integral in relation to the 3 axes

Absolute value of the integral (Trapezoidal method)				
		Angular Errors		
		Positions		
Controller	Axis X	Axis Y	Axis Z	Total
PD	78.1237	84.5526	31.1589	193.8354
PD-GA	43.1680	12.6005	7.1152	62.8839
CN-GA	5.3271	4.1397	2.0544	11.5214
		Speed		
Controller	Axis X	Axis Y	Axis Z	Total
PD	0.93837	0.32763	0.29725	1.56326
PD-GA	0.61499	0.44636	0.38184	1.44321
CN-GA	1.42073	1.31787	0.56772	3.30633

The controllers obtained are robust, and the results produced validate the use of genetic algorithms as an optimization tool to treat similar problems as those addressed in this work.

References

Guerra, R.; Sandri, S. A.; Souza, M. L. O, 1997, "Dynamics and design of autonomous attitude control of a satellite using fuzzy logic". *Anais do Congresso Brasileiro de Engenharia Mecânica*, Bauru, Brazil, n. COB 1338. (COBEM'97).

Souza, L.C.G.and Silva,A.R., 1999,"French-Brazilian Micro-Satellite Control System Design During Normal Mode". In Applied Mechanics in the Americas, Vol, 8, Ed. H. I. Webwer, P.B. Gonçalves, I. Jasiuk, D. Pamplona, C. Steele and L. Bevilacqua, pp. 1163-1166, Published by AAM and ABCM – Rio de Janeiro, ISBN: 85-900726-2-2.

Silva, A. R. da, 1997, "Estudo de sistema de controle de um satélite artificial durante a fase de transferência orbital e apontamento". *Dissertação de Mestrado em Eng. Espacial e Tec. Espaciais/Mecânica Espacial e Controle.* Instituto Nacional de Pesquisas Espaciais, São José dos Campos, Brasil, INPE-6397-TDI/613, 86 p.

Driankov, D.; Hellendoorn, H.; Reinfrank, M., 1993. *"An Introduction to Fuzzy Control"*. Springer-Verlag.

Lee, C. C., 1990, "Fuzzy logic in control systems: Fuzzy logic controller (part I)". *IEEE Transactions on Systems, Man an Cybernetics*, v. 20, n. 2, p. 404-418.

Woodward, M. A., July 1996, "Fuzzy open-loop attitude control for the FAST spacecraft". *Proc. of the NASA AIAA, Guidance, Navigation and Control Conference*, San Diego.

Conway, D.; Sperling, R.; Folta, D.; Richon, K.; DeFazio, R., 1994. "Automated maneuver planning using a fuzzy logic controller". *Proc. of the NASA AIAA, Guidance, Navigation and Control Conference*, San Diego.

Guerra, R.; Sandri, S. A.; Souza, M. L. O., "Controle de atitude autônomo de satélites usando lógica nebulosa". *III Simpósio Brasileiro de Automação Inteligente*, Vitória, Brazil, p. 337-342, 1997. (SBAI'97).

Jang, J. S. R., 1993. "ANFIS: Adaptive-network-based fuzzy inference system". *IEEE Transactions on Systems, Man an Cybernetics*s, v. 23, n. 3, p. 665-685.

Lin, C. T., 1995, "A neural fuzzy control system with structure and parameter learning". *Fuzzy Sets and Systems*, n. 70, p. 183-212.

Husbands, P., 1992, "Genetic algorithms in optimization and adaptation". In: Kronsjo, L.; Shunsheruddin, D., ed. *Advances in paralell algorithms*. Blackwell Scientific Publications, p. 227-277.

Karr, C.,1991, "Applying genetics to fuzzy logic". *AI Expert*, p. 38-43.

Karr, C., 1994, "Adaptive control with fuzzy logic and genetic algorithms". In: Yager, R. R.; Zadeh, L. A., ed. *Fuzzy sets, neural networks and soft computing*. Van Nostrand Reinhold, p. 345 – 367.

Pham, D. T.; Karaboga, D., 1991, "Optimum design of fuzzy logic controllers using genetic algorithms". *J. Syst. Eng.*, v. 1, p. 114-118.

Kinzel, J.; Klawonn, F.; Kruse, R., 1994, "Modifications of genetic algorithms for designing and optimizing fuzzy controlers". *Proc. of the IEEE I Int. Conf. on Evolutionary Computation*, Orlando, p. 28-33.

Kim, J.; Moon, Y.; Zeigler, B. P., 1994, "Designing Fuzzy Net Controllers Using GA Optimization". *Proc. of IEEE/IFAC Joint Symposium on Computer-Aided Control System Design. Tucson*.

Mohamadian, M.; Stonier, R. J., 1994, "Generating fuzzy rules by genetic algorithms". *Proc. of the IEEE III Int. Workshop on Robot and Human Communication*, Nagoya, p. 362-367.

Mohamadian, M.; Stonier, R. J., 1994, "Tuning and optimization of membership of fuzzy logic controlers by genetic algorithms". *Proc. of the IEEE III Int. Workshop on Robot and Human Communication*, Nagoya, p. 356-361.

Aya, J. C. C.; Costa, O. L. V., 1997, "Otimização de controladores nebulosos usando algoritmos genéticos". *III Simpósio Brasileiro de Automação Inteligente*, Vitória, p 207-212. (SBAI'97).

Silva, J. C. A.; Kierner, C., 1997, "Algoritmo genético aplicado a simulação". *III Simp\osio Brasileiro de Automação Inteligente*, Vitória, p. 283- 288. (SBAI'97).

Vasconcelos, J. A. D.; Neiba, A. M.; Cardoso, E. N.; Pinheiro, F. A., 1997, "Algoritmo genético aplicado ao controle de tensão em sistemas elétricos de potência". *III Simpósio Brasileiro de Automação Inteligente*, Vitória, p. 38-44.

Hoffmann, F., May 1997, "Genetic algorithm". [online] <http://www.cs.berkeley.edu:80/~fhoffman/ga.html>.

Zadeh, L. A. ,1965, "Fuzzy sets, information and control". v. 8. p. 338-353.

Mamdani, E. H., 1976, "Advances in the linguistic syntesis of fuzzy controllers". *Int. J. Man-Mach. Stud*, v. 8, p. 669-678.

Mamdani, E. H.; Baaklini, N., 1975, "Prescriptive method for deriving control policy in a fuzzy logic controller". *Eletronic Letters*, v. 11, p. 625-626.

Mamdani, E. H.; Procyk, T.; Baaklini, N., "Application of fuzzy logic to controller design based on linguistic protocol". In: Mamdani, E. H.; Gaines, B. R., ed. Discrete systems and fuzzy reasoning. London: University of London. p. 125 - 149.

Bauer, P.; Nouak, S.; Winkler, R., Apr. 1998, "A brief course in fuzzy logic and fuzzy control". [online]. <ftp://ftp.flll.uni-linz.ac.at/pub/info/flllorg.ps.gz>.

Guerra, R., 1998, "Projeto e simulação do controle de atitude autônomo de satélites usando lógica nebulosa". *Dissertação de Mestrado em Eng. Espacial e Tec. Espaciais/Mecânica Espacial e Controle*. Instituto Nacional de Pesquisas Espaciais, São José dos Campos, INPE-6714-TDI/630, 190 p.

Heider H., Tryba V., Mühlenfeld E., 1995, "Automatic design of fuzzy systems by genetic algorithms". In: Bouchon-Meunier B., Yager R. R., Zadeh L. A., ed. *Fuzzy Logic and Soft Computing*. World Scientific, p. 21-28.

Larsen, P. M., 1998, "Industrial aplications of fuzzy logic control". In: Mamdani, E. H.; Gaines, B. R., ed. *Fuzzy Reasoning and its Aplications*. London: Academic Press Inc., p. 335 – 342.

Sanchez, E.; Shibata, T.; Zadeh, L. A., 1997, *"Genetic algoritms and fuzzy logic systems: soft computing perspectives"*. Singapore: World Scientific Publishing Co. Pte. Ltd., 240 p.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 495-506

ISSN 0100-7386
Printed in Brazil

Towards a Bapta Mechanism for Small Satellites

Mário César Ricci
Sebastião Eduardo Corsatto Varotto
National Institute for Space Research – INPE
Space Mechanics and Control Division - DMC
12227-010 São José dos Campos, SP Brazil
mcr@dem.inpe.br
varotto@dem.inpe.br

Abstract

This work intends to show some aspects about the mechanical layout and modeling of a BAPTA (Bearing and Power Transfer Assembly) mechanism in development involving a government institution (INPE) and private companies. The design here described is being evolved for application in small satellites (~0.5 kW with a rigid solar panel of ~1.0 kgm^2). The selected concept uses a conventional 1.8° stepper motor and a 100:1 harmonic drive reduction gearing to achieve a theoretical output step size of 0.018°. Simulations in computer are showing that significant interaction with the solar array and spacecraft can be avoided (spacecraft pitch disturbing velocity is lower than 10^{-3} deg/sec) with power consumption as low as 3 to 5 W. The adequate determination of the power consumption is possible due to a adequate representation of magnetic non-linearity in the motor, including its effect on electromagnetic torque production. The reliable prediction of the important dynamic characteristics of the motor and general system necessitates a precise representation of the flux-linkage data. A good example is the single-step damping, which can be predicted satisfactorily only if the current disturbance and associated power loss in each stator circuit are calculated correctly. This requires that the rate of change of flux-linkage be defined accurately for any combination of the system variables.

Keywords: *BAPTA, Small Satellites, Modeling.*

Introduction

Some kinds of artificial satellites must be appointed toward the Earth while the solar panels must be appointed toward the Sun in order of maximize the solar energy that reaches the solar panels. These kinds of satellites shall incorporate a BAPTA (Bearing and Power Transfer Assembly) mechanism.

The BAPTA is a Power Subsystem mechanism that performs the tasks of maintain the one degree-of-freedom of the output axis (panel axis) while transmits the electrical and power signals from the panel to the satellite body. The BAPTA consists basically of three main units: a) the Bearing Unit, b) the Drive Unit and c) the Slip Rings Unit.

The design here described is being evolved for application in small satellites with a rigid solar panel of about 0.5 kW and 1.0 kgm^2 of inertia. The platform inertia shall be not less than 20 kgm^2.

In this work we concentrate the efforts in the modeling of the Drive Unit. Two solutions were considered for the Drive Unit: a) closed-loop synchronous system and b) open-loop incremental motion control system. Since the orbital period is too large (on the order of 10^2 min) the nominal speed and its variation are very low and very difficult to measuring. Therefore, the second solution is to be the better choice and the selected concept uses a conventional 1.8° stepper motor and a 100:1 harmonic drive reduction gearing to achieve a theoretical output step size of 0.018° (Fig. 1).

In normal mode the mechanism operates in open-loop with a frequency of about 3.11 steps per second. When the solar panel's normal deviates from Satellite-Sun vector of an angle greater than two degrees a signal is send to the control electronics to modify the system's clock in order to maintain the solar panel's pointing requirements.

The main specifications for the BAPTA mechanism, operating in a satellite with a orbital period of 107 minutes, are given in Table 1.

Table 1 Main Specifications for the BAPTA Mechanism

Orbital mean rate (orbital period of 107 min)	0.056 o/s
Pitch disturbing velocity	< 10-3 o/s
Solar panel's inertia	1 kg m2
Platform inertia	20 kg m2
Total mass	< 2 kg
Power consumption	5 W
Overall dimensions	
Electronic box	120x120x120 mm
Mechanical part	120x120x120 mm

Equations of Motion

The system under investigation consists of a rigid asymmetric platform (with inertia J_1 and attitude θ_1) linked to an axisymmetric body (with inertia J_2 and attitude θ_2), stands for the motor's rotor and reduction gearing, that is connected by a lossy torsional spring to a third axisymmetric body (with inertia J_3 and attitude θ_3), representing the rigid solar panel and the yoke as illustrated in Fig. 2.

Disregarding the translational motion of the spacecraft the degrees-of-freedom of the system are θ_1, θ_2 *and* θ_3, where the attitude θ_1 is to be controlled with respect to an absolute reference and the masses are free to oscillate about the common axis of symmetry. The equations of motion may be arranged into eight 1st-order differential equations:

$$\frac{d\theta_1}{dt} = \omega_1, \tag{1}$$

$$\frac{d\omega_1}{dt} = -\frac{1}{J_1}\left[T(i_{AC},\theta_{21}) + T(i_{BD},\theta_{21}) - k_v\omega_{21} - T_f\frac{\omega_{21}}{|\omega_{21}|}\right], \tag{2}$$

$$\frac{d\theta_2}{dt} = \omega_2, \tag{3}$$

$$\frac{d\omega_2}{dt} = \frac{10^4}{J_2}\left[T(i_{AC},\theta_{21}) + T(i_{BD},\theta_{21}) - k_v\omega_{21} - T_f\frac{\omega_{21}}{|\omega_{21}|}\right] + \frac{1}{J_2}\left(k\theta_{32} + c\omega_{32}\right), \tag{4}$$

Nomenclature

c	=	equivalent damping coefficient,
k	=	spring constant,
κ_v	=	coefficient of viscous friction,
T_f	=	friction torque,
T	=	electromagnetic torque,
i	=	motor current,
J	=	moment of inertia,
N_r	=	number of rotor teeth,
R	=	total series resistance of a stator circuit,
V	=	voltage applied,
W'	=	co-energy,

Greeks:

θ	=	instantaneous angular position,
λ	=	flux-linkage,
ω	=	instantaneous angular velocity,

Subscripts:

a	=	refers to the change in flux-linkage of phase A as a function of i_{AC} and θ,
A	=	refers to the motor's phase A,
B	=	refers to the motor's phase B,
C	=	refers to the motor's phase C,
D	=	refers to the motor's phase D,
AC	=	refers to the current difference between the motor's phases A and C,
BD	=	refers to the current difference between the motor's phases B and D,
PMA	=	refers to the change in flux-linkage of phase A due to the permanent magnet as a function of the angular position,
1	=	refers to the satellite body[1]
2	=	refers to the motor's rotor and reduction gearing input and output axis,
3	=	refers to the solar panel and yoke,
21	=	refers to the position and velocity displace-ments between the rotor and the platform,
32	=	refers to the position and velocity displace-ments between the solar panel plus yoke and the reduction gearing output axis.

$$\frac{d\omega_3}{dt} = -\frac{1}{J_3}\left[k\theta_{32} + c\omega_{32}\right], \tag{5}$$

$$\frac{d\theta_3}{dt} = \omega_3, \tag{6}$$

$$\frac{di_{AC}}{dt} = \frac{1}{\dfrac{\partial\lambda_A}{\partial i_{AC}}}\left[\frac{V_A - V_C}{2} - \frac{Ri_{AC}}{2} - \omega_{21}\frac{\partial\lambda_A}{\partial\theta_{21}}\right], \tag{7}$$

$$\frac{di_{BD}}{dt} = \frac{1}{\dfrac{\partial\lambda_B}{\partial i_{BD}}}\left[\frac{V_B - V_D}{2} - \frac{Ri_{BD}}{2} - \omega_{21}\frac{\partial\lambda_B}{\partial\theta_{21}}\right], \tag{8}$$

where (1) and (2) are the equations for satellite body; (3) and (4) are representing the rotor axis and reduction gearing; (5) and (6) refers to the solar panel plus yoke and (7) and (8) are the electromagnetic equations for motor currents as described as follow.

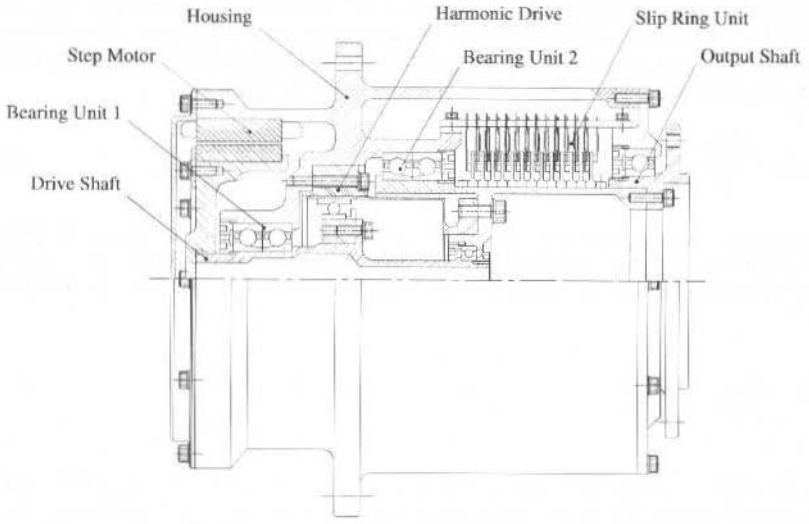


Fig. 1 BAPTA mechanism for small satellites

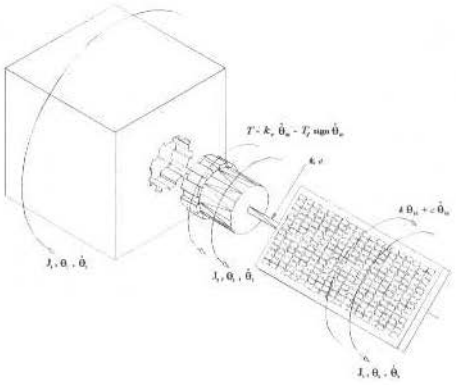


Fig. 2 A simplified model for a small satellite with a rigid solar panel

Motor's Model

General Considerations

The motor under consideration here is a low-speed, two phase synchronous inductor motor as described by Snowdon and Madsen (Snowdon, et al., 1962) (Ricci, 1986). The stator has eight salient poles which are wound with a 2-phase, 4-pole winding. Each salient pole possesses five teeth having a pitch of one forty-eighth of a revolution. The rotor is constructed of two stacks, each having 50 equally spaced teeth, separated by a axially-magnetized permanent magnet and the teeth on the rotor stacks offset by half a rotor tooth pitch.

To reduce the complexity of the drive circuits the motors are often provided with bifilar windings. The motor may be regarded then as having four independent phases and, instead of reversing the current in a winding, current of the original polarity is supplied to a bifilar coil connected in the opposite sense. This is the arrangement employed in the motor which is the subject of the present work.

With bifilar windings the motor can be thought as a four phase machine and these will be referred to as A, B, C and D respectively. Phases A and C are the two components of one bifilar winding and phases B and D are the two components of the other bifilar winding. With this arrangement, 100% coupling exists between two phases comprising a bifilar pair and currents of the same polarity flowing in the phases A and C, for instance, produce opposing magnetization of the core. Because the quadrature between the two bifilar windings the mutual coupling between them can be considered negligible.

The basic electrical circuit equation, taking phase A as the example, is

$$V_A = Ri_A + \frac{d\lambda_A}{dt}. \tag{9}$$

Considering the coupling between phases A and C (and between the phases B and D) the four differential equations for the electrical circuits are not independent. There are, in fact, just two independent differential equations, one for each bifilar pair of windings. For the bifilar-pair A-C the equation is

$$V_A - V_C = Ri_{AC} + 2\frac{d\lambda_A}{dt}. \tag{10}$$

The flux-linkage λ_A may be expressed as $\lambda_A(i_{AC},\theta)$. A similar $\lambda_B(i_{BD},\theta)$ exists, of course, appropriate to the bifilar-pair B-D. Substituting for λ_A and λ_B, and combining the mechanical equations with those of the electrical circuits, yields the eight 1st-order differential Eqs. (1-8).

The expression for the flux-linkage of each pair of bifilar coils, in terms of net coil current i and the angular position of the rotor θ, may be determined according to Pickup and Russel (Pickup, et al., 1980), by fitting mathematical functions to experimental flux-linkage data obtained from static tests. As indicated previously, in order to use the model for prediction of the dynamic response, it is necessary to be able to specify $\partial\lambda/\partial i$, $\partial\lambda/\partial\theta$ e $T(i,\theta)$, given instantaneous values of the variables i and θ.

Taking phase A, for example, the flux-linkage can be expressed as

$$\lambda_A(i_{AC},\theta)=\lambda_{PMA}(\theta)+\lambda_a(i_{AC},\theta). \tag{11}$$

Here, $\lambda_{PMA}(\theta)$ is a flux-linkage of phase A due to the permanent magnet flux when the net current in bifilar pair A-C is zero. $\lambda_a(i_{AC},\theta)$ represents the change in flux linkage of phase A which occurs owing to the flow of a net current i_{AC} in the coils of that phase.

Provided that $\lambda_{PMA}(\theta)$ and $\lambda_a(i_{AC},\theta)$ are continuous functions, $\partial\lambda_A/\partial\theta$ and $\partial\lambda_A/\partial i_{AC}$ can be obtained directly by differentiation of Eq. (11). The electromagnetic torque $T(i_{AC},\theta)$ can be calculated from the magnetic co-energy $W'(i_{AC},\theta)$. The co-energy is given by

$$W'(i_{AC},\theta)=\int_0^{i_{AC}}\lambda_A\left(i'_{AC},\theta\right)di'_{AC}, \tag{12}$$

where i'_{AC} is a dummy variable.

Substituting Eq. (11) into Eq. (12) yields

$$W'(i_{AC},\theta)=\lambda_{PMA}(\theta)i_{AC}$$

$$+\int_0^{i_{AC}}\lambda_a\left(i'_{AC},\theta\right)di'_{AC}. \tag{13}$$

The torque produced is then given by

$$T(i_{AC},\theta)=\frac{\partial W'}{\partial\theta}(i_{AC},\theta)\Big|_{i_{AC}=cte.}. \tag{14}$$

Substituting Eq. (13) for the co-energy, into Eq. (14), and performing the differentiation, gives

$$T(i_{AC},\theta)=i_{AC}\frac{d\lambda_{PMA}(\theta)}{d\theta}$$

$$+\frac{\partial}{\partial\theta}\int_0^{i_{AC}}\lambda_a\left(i'_{AC},\theta\right)di'_{AC}. \tag{15}$$

Thus, the torque consists of two components. The first term of Eq. (15), can be thought of loosely as the torque produced by the interaction of the permanent magnet flux and the coil current i_{AC}. Similarly, the second term, may be considered to result from the interaction of the current-dependent flux-linkage and the coil current.

The derivatives of the flux-linkage of phase B, $\partial\lambda_B/\partial\theta$ and $\partial\lambda_B/\partial i_{BD}$, and the torque $T(i_{BD},\theta)$ from bifilar pair B-D, can be calculated, of course, in precisely the same manner.

Mathematical Representation of Flux-linkage

Permanent Magnet Flux-linkage

he flux-linkage of phase A from the permanent magnet is symmetrical about the stable zero-torque position ($N_r\ \theta = 0°$) and the unstable zero-torque position ($N_r\ \theta = 180°$), and is equal and opposit about ($N_r\ \theta = 90°$), with N_r being the number of rotor teeth. Thus, the following function (with n an odd integer) is suitable for $\lambda_{PMA}\ \theta$:

$$\lambda_{PMA}(\theta) = \lambda_1 \cos(N_r\theta) + \lambda_3 \cos 3(N_r\theta) + \cdots$$

$$\cdots + \lambda_n \cos n(N_r\theta). \qquad (16)$$

To obtain the experimental flux-linkage readings, taken with zero stator excitation, a stable zero torque position for phase A was located and this was employed subsequently as the reference position. With the machine unexcited and the rotor clamped, the stator was rotated through a given angle and the associated change of flux-linkage recorded on a fluxmeter connected across the windings of phase A (or C). Having recorded the value of flux-linkage, the stator was returned to its reference position and the fluxmeter reset to zero. The test was repeated then for a different value of angular displacement. A similar flux-linkage characteristic can be obtained for phase B, with the reference position displaced by.$1.8(N_r\ \theta = 90°)$

To fit Eq. (16) directly to the permanent magnet data, a coefficient λ_0 must be included in the expression to allow for the fact that the reference (zero) level of flux-linkage has been taken at value of ($N_r\ \theta = 0°$) where the absolute flux-linkage of phase A is at a maximum value (obvioulsy this procedure is not necessary if ($N_r\ \theta = 90°$), the position where, by simmetry, the absolute value of permanent-magnet flux-linkage of phases A and C is zero, is taken as the reference from wich the change of flux-linkage is measured).

The coefficients of Eq. (16) were determined from the experimental points using the least-square-error method of curve fitting. It was found that the contend of harmonics above the third is negligible and, therefore, the following function can be taken to represent adequately the flux-linkage data

$$\lambda_{PMA}(\theta) = \lambda_1 \cos(N_r\theta) + \lambda_3 \cos 3(N_r\theta). \qquad (17)$$

This function gives an average discrepancy per experimental point of $0.14x10^{-3}$ Vs, which is comparable with the experimental accuracy of measurement. The apropriate coefficients of equation are $\lambda_1 = 7.45x10^{-3}$ Vs and $\lambda_3 = 7.45x10^{-5}$ Vs. The Fig. 3 shows the experimental measurements and theoretical representation from curve fit for the permanent magnet flux-linkage.

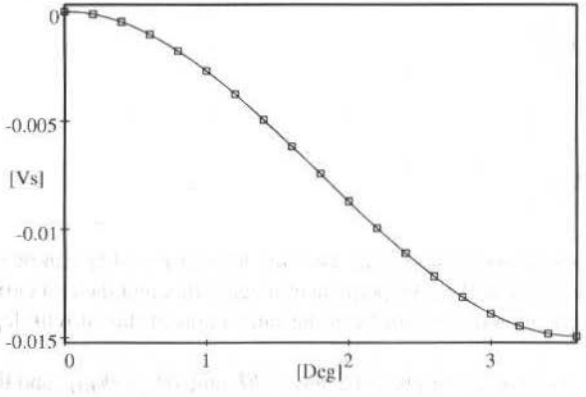


Fig. 3 Permanent magnet flux-linkage. Experimental and theoretical values

Current-dependent Flux-linkage

The current-dependent linkage of phase A was measured by using the bifilar-wound coils of phase C as a closely-coupled search coil connected to a fluxmeter. Because of the equal number of turns on phases A and C this arrangement gives a direct reading of the flux-linkage changes which occur in phase A.

Initially the rotor and stator were set up in the reference position ($N_r\ \theta = 0°$) as defined in previous sub-section. A positive current of 0.5 A was switched on in phase A and the associated change in the flux-linkage was recorded on the fluxmeter connected to the bifilar winding. This reading gives the current-dependent flux-linkage at $N_r\ \theta = 0°$ for 0.5 A of positive excitation. The current-dependent flux-linkage at other positions of rotor relative to stator was obtained in a similar manner. The complet test was then repeated for several different values of positive phase current.

The flux-linkage can be modelled by fitting a mathematical function to the experimental current-dependent data. The flux-linkage corresponding to negative values of current can be calculated directly from this function, as will be indicated later.

A general expression for the flux-linkage, bearing in mind that $\lambda_a\ (i_{AC},\ \theta)$: must be an odd function of i_{AC} and be symmetrical in $N_r\ \theta$ about ($N_r\ \theta = 0°$) and ($N_r\ \theta = 180°$), is

$$\lambda_a(i_{AC},\theta)=\sum_{r=0}^{k}\left(B_{1r}i_{AC}+B_{3r}i_{AC}^{3}+\cdots\right.$$

$$\left.\cdots+B_{jr}i_{AC}^{j}\right)cosr(N_r\theta), \tag{18}$$

where j is an odd integer.

The optimum values of the $[(j+1)/2](k+1)$ coefficients for a given combination of j and k can be determined using an adapted version of least-square-error method. It was found that, on the bases of economy of computing and accuracy of representation, it is necessary to have j=5 and k=2. Further increasing the degree of the i_{AC} polynomials improves the accuracy marginally. Harmonics in $N_r\ \theta$ above the second are negligibly small. The Table 2 shows the coefficients obtained by the least-square method and the Fig. 4 shows the current-dependent flux-linkage fitting as a function of angular position.

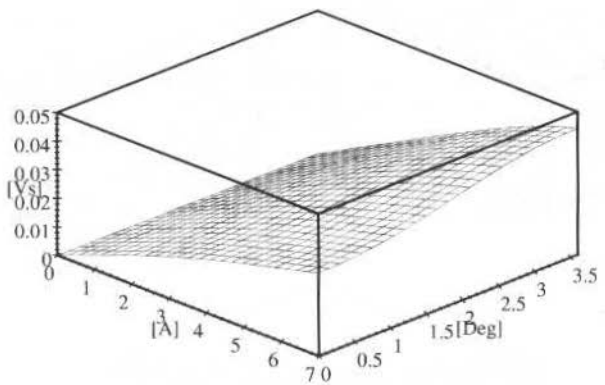


Fig. 4 Current-dependent flux-linkage fitting

The chosen function gives an average discrepancy per experimental point similar to that obtained for the permanent-magnet flux-linkage. The flux-linkage for negative values of i_{AC} in phase A (which is equivalent to positive current in phase C) are obtained by substituting the appropriate i_{AC} into Eq. (18) and replacing $N_r\ \theta$ by $N_r\ \theta - \pi$.

Aditional tests were performed to determine whether or not coupling between the bifilar pairs A-C and B-D was significant. Various levels of direct current excitation were applied to phase B and the measurements of current-dependent flux-linkage in phase A were repeated. It was found that at highest level of excitation used (7 A) the readings obtained were virtually unaffected by the presence of the current in phase B. It can be assumed, therefore, that the common return flux paths in the back of stator core were not significantly magnetically non-linear at this level of stator excitation.

Table 2 Current-dependent flux-linkage fitting coefficients

r	$B_{1r}\times10^{-3}$	$B_{3r}\times10^{-5}$	$B_{5r}\times10^{-7}$
0	6.234470	-2.49055	0.858076
1	-0.917669	-1.94243	3.29916
2	-0.170484	0.925831	-1.10692

Static Torque/Angle Characteristics

In order to solve the simultaneous dynamics Eqs. (1-8), the electromagnetic torque of the machine is required in terms of instantaneous values of the coil currents and angular position of the rotor. This can be calculated from the flux-linkage. With the functions $\lambda_{PMA}(\theta)$: and $\lambda_a(i_{AC}, \theta)$: derived previously, the torque expression for a single phase becomes

$$T(i_{AC},\theta)=-N_r\{i_{AC}[\lambda_1\sin(N_r\theta)+3\lambda_3\sin3(N_r\theta)]$$

$$+\sum_{r=1}^{2}r\left(B_{1r}\frac{i_{AC}^2}{2}+B_{3r}\frac{i_{AC}^4}{4}+B_{5r}\frac{i_{AC}^6}{6}\right)\sin r(N_r\theta)\}. \qquad (19)$$

Thus, the two torque components calculated from Eq. (15) can be evaluated to yield the static torque/angle characteristics at various levels of stator excitation. The Figures 5(a) and 5(b) displays the two torque components using the flux-linkage coefficients. The sum of the components is plotted in Fig. 5(c).

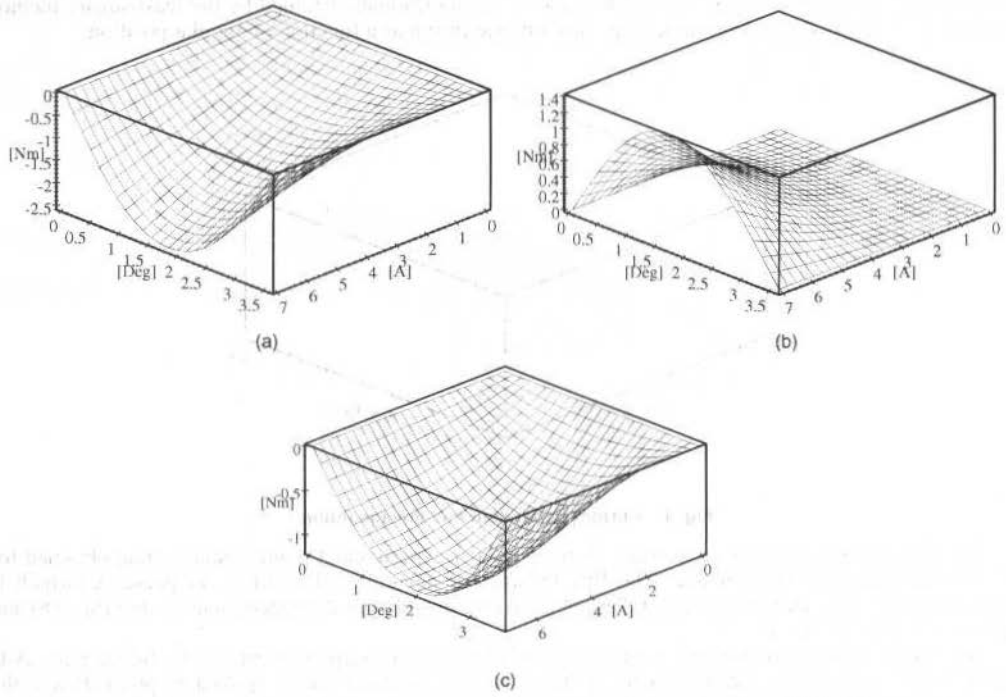


Fig. 5 Static Torque/Angular Position Characteristics. Due to (a) $i_{AC}d\lambda_{PMA}(\theta)/d\theta$ **; (b)** $\frac{\partial}{\partial\theta}\int_0^{i_{AC}}\lambda_a(i_{AC},\theta)di_{AC}$ **; (c) the sum of the two components**

Numerical Results

Some preliminary results are shown in this section. Since it is the qualitative behavior of solutions that is of interest, the results are presented here in graphical form. The computations were performed using a software for simulating dynamic systems named SIMULINK™ that is an extension to MATLAB™. SIMULINK works with windows called *block diagram*. In these windows the model is created and edited by mouse commands. After definition of the model it can be analysed by simulation algorithms and the progress of the simulation can be viewed while the simulation is running. The results can be made available in the MATLAB workspace when the simulation is complete.

The following system's parameters are assumed for the model (1-8) as shown in Table 3.

Table 3 System's parameters

J_2	20.0 Kgm2
J_2	0.01 Kgm2
J_3	0.8 Kgm2
k	125.0 Nm/rd
c	0.1 Nms/rd
T_f	0.05 Nm
k_v	0.0005 Nms/rd
V	30 V
R	200 Ω

The results of simulation are shown in Figs. 6-13 and are appropriate to 2-phase excitation using the switching sequence indicated in Table 4.

Table 4 Sequence of excitation for hybrid motor

Step	Phase A	Phase B	Phase C	Phase D
1	On	On	Off	Off
2	On	Off	Off	On
3	Off	Off	On	On
4	Off	On	On	Off

Figures 6 and 7 show the platform angular velocity. It can see that the spacecraft pitch disturbing velocity is on order of 10^{-5} deg/sec.

Figures 8 and 9 show the net current i_{AC}. The same magnitude inverted profile it was found to i_{BD}. Fig. 10 shows the driver's power consumption es-timate in watts/phase. It is possible to decrease these values by a factor of three (and even more) by increas ing R or switching off the motor current after the settling time has been achieved. The use of the model can be extended to study the effects of using dual-volt- age supplies or chopped excitation too.

Figure 11 shows the motor torque $T(i_{AC}, \theta)$. The same inverted profile it was found to $T(i_{BD}, \theta)$.. It can observe an average motor torque of about 0.06 Nm under normal operation.

Finally, Figs. 12 and 13 show the relative motor's rotor and solar array angular displacements as having the spacecraft as a reference.

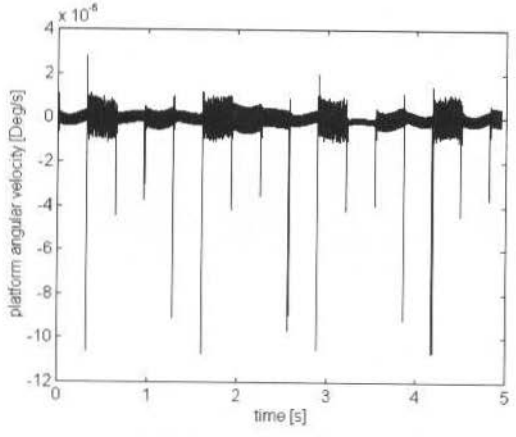


Fig. 6 Spacecraft pitch disturbing velocity

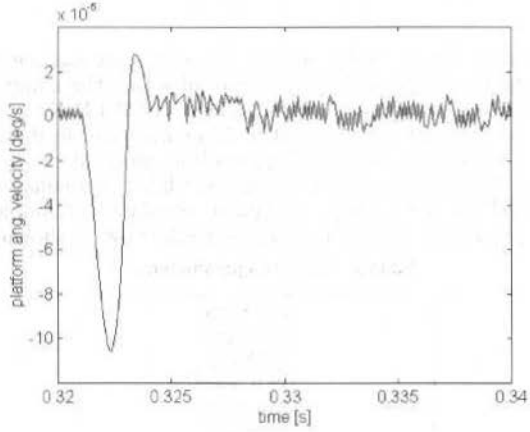


Fig. 7 Spacecraft pitch disturbing velocity (detail)

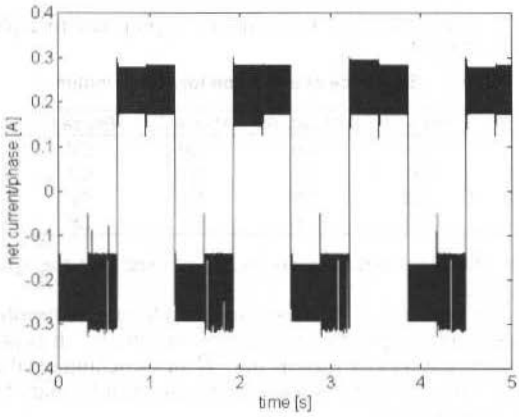


Fig. 8 Net current i_{AC}

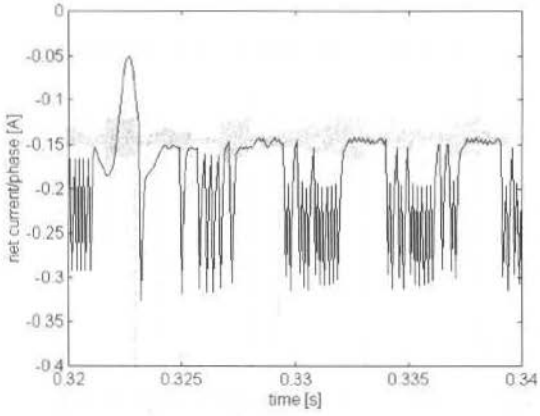


Fig. 9 Net current i_{AC} (detail)

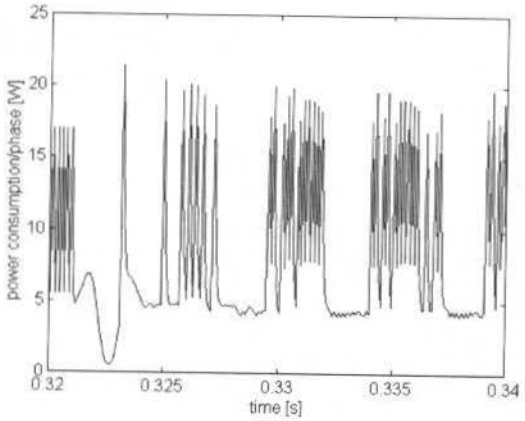


Fig. 10 Power consumption/phase (detail)

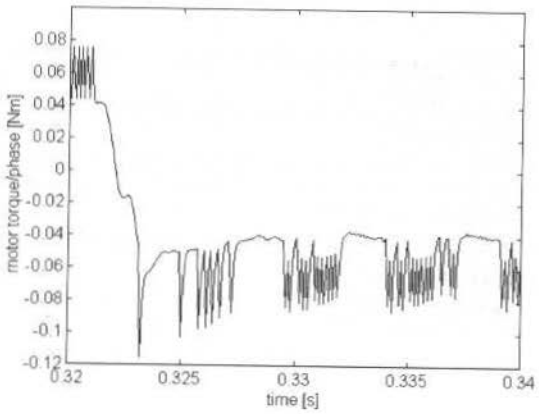


Fig. 11 Motor torque $T(i_{AC},\theta)$ (detail)

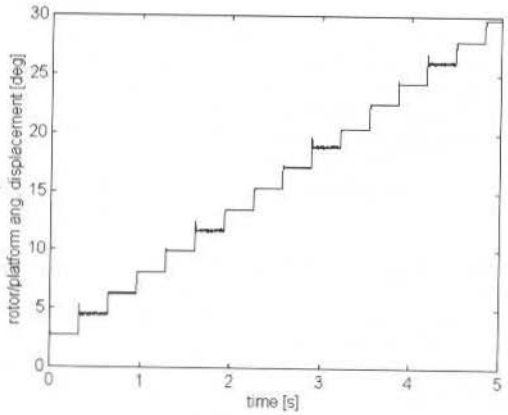


Fig. 12 Motor's rotor/platform angular displacement

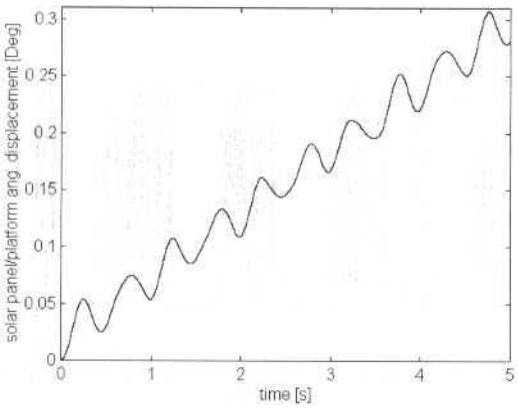


Fig. 13 Solar panel/platform angular displacement

References

Snowdon, A. E., Madsen, E. W., 1962, "Characteristics of a Synchronous Inductor Motor". *Trans. Amer. Inst. Elect. Engrs*, IGA-81, pp. 1-5.

Ricci, M. C., Ago. 1986, "Motor Síncrono de Indução: Características Construtivas e Operacionais". São José dos Campos, INPE, 42 p. (INPE-3976-RPE/517).

Pickup, I. E. D.; Russel A. P., 1980, "A Model for Predicting the Dynamic Characteristics of Hybrid (Permanent Magnet) Synchronous/Stepping Motors". *Proceedings of 9th Annual Symposium on Incremental Motion Control Systems and Devices*, University of Illinois, pp. 1-13.

Simulink A Program for Simulating Dynamic Systems. *User's Guide*, The MathWorks, Inc.

Matlab *User's Guide*, The MathWorks, Inc.

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 507-513

ISSN 0100-7386
Printed in Brazil

Star Catalogue Facility

Alan Batten
Science Systems (Space) Ltd.,
23 Clothier Road, Bristol BS4 5SS
alan.batten@scisys.co.uk

Abstract

The provision of a star catalogue is often a necessary requirement for the support of a mission. In the past the European Space Operations Centre (ESOC) has used dedicated star catalogues prepared on an individual basis to support such missions as Exosat, Hipparcos and ISO. However for future missions it is preferred to develop a central facility, which can provide the catalogues and tools to support the missions, at the same time providing a standardisation of the handling of star data across the missions.

The Star Catalogue Facility (SCF) is developed to meet this need and was designed based on the experience of recent missions and the requirements of missions planned for the first decade of the next century. It provides a database of celestial objects for the production of star catalogues to support individual missions and the tools to handle the star data.

Central to SCF is the master catalogue, which is built by combining the star data from a number of different sources: primarily the outputs from the Hipparcos mission and the Hubble Guide Star Catalogue. Mission specific catalogues are generated from the master catalogue. Each mission catalogue is tailored to the specific requirements of the particular mission, for example the restriction to include objects within a certain magnitude range. Also mission specific parameters such as a guide star classification and/or an instrumental magnitude may be added to the star record. In addition SCF provides standard for handling catalogues, such as field of view plotting, generation of density maps and generation of statistics. Within the context of the Star Catalogue Facility project, the opportunity was also taken to develop generic attitude software for the support of missions utilising star sensors

This paper describes the central features of the Star Catalogue Facility and the influence of earlier missions on the SCF design. The paper then demonstrates how SCF has been applied for the Flight Dynamics support of future missions such as XMM, Envisat, Integral and Rosetta

Keywords: Star Catalogue, attitude software

Introduction

Earlier ESA missions such as EXOSAT, HIPPARCOS, ISO and GIOTTO requiring star data (position, magnitude, proper motion etc.) for on ground operations utilised dedicated star catalogues prepared on an ad-hoc basis either within or outside ESA.

Exosat:

Around the time of preparation for the Exosat mission in the mid-seventies the requirement for a general Star Catalogue Facility was first realised. However, as development of this first star catalogue facility was in parallel with Exosat software development a catalogue to support the Exosat mission was compiled from astrometric and photometric source catalogues provided by the Centre de Donnees Stellaires (CDS) on an ad-hoc basis.

To allow for efficient access to stars in a particular area of the celestial sphere consideration has to be given as to how to organise the star catalogue data. Techniques usually involve dividing up the celestial sphere into areas, which are not necessarily equal, and indexing these areas. The structure for the Exosat star catalogue was to partition that data according to a **celestial cube** structure (RD1) and to organise the catalogue as a direct access file. The celestial cube method first partitions the sphere into 6 equal areas or **zones** defined by the projections of the faces of a cube. Each of these is then further subdivided into N*N areas, where the parameter N is referred to as the **partitioning factor**.

Hipparcos:

The Input Catalogue (INCA), together with a number of annexes, was provided to ESOC by the Hipparcos INCA consortium. From these catalogues an operational catalogue was developed according to the celestial cube structure with various mission specific parameters computed, such as the target observation times, lower and upper minimum observation times. A further off-line process ran daily to derive an 'extended programme star file', which contained data required for uplink to the on-board programme star file and also data required for on-ground payload monitoring software. Special programme star files were also generated to support initial calibration activities.

ISO:

For ISO a contract was placed with CDS to provide a catalogue based on the CDS databases and, when available, results from the Hipparcos data reduction. This catalogue contained certain mission specific features; such as a guide star classification scheme based on proximity to disturbing stars according to criteria derived from the properties of the ISO start tracker. This catalogue was processed to produce an operational star catalogue, again structured according to the celestial cube. An instrumental magnitude was derived and the star position propagated to an appropriate epoch. This catalogue was regenerated quarterly.

Star Catalogue Facility Design

To standardise the approach to on ground handling of star data it was decided to develop a generic Star Catalogue Facility (SCF) to support future missions.

The facility allows the generation of a master catalogue from a set of source catalogues to serve as a unique reference for generating the mission specific catalogues.

Additionally SCF provides tools for handling catalogues such as graphics and generation of density maps.

SCF also handles ephemeris data for a selected range of solar system objects.

SCF is written in FORTRAN to run on a SUN workstation environment. The man-machine interfaces (MMIs) were developed using the GUItool application from ESOC's ORATOS infrastructure. Graphics were developed using TCL/TK.

Catalogue Data Organisation

The celestial cube method of organising star catalogues used in the earlier ESA mission had shown good performance for operational support and was therefore adopted as the standard for SCF.

For the earlier missions the structure of the celestial cube catalogue was contained within the actual catalogue file as header records. The approach adopted within SCF is to describe the structure of the celestial cube catalogue in a second file: the catalogue definition file. The definition file lists the partitioning factor along with the list of the number of stars in each zone of the cube - this allows for fast access of stars referenced by their position on the celestial sphere. The definition file also lists other general data on the catalogue such as the reference system, epoch equinox and the data fields contained within the catalogue.

Moreover the concept of a definition file is extended to allow the handling of catalogues imported from other sources. For example the definition file for an ASCII catalogue contains a description of which data is contained in the catalogue, which bytes of the input record the data occupies and the units of that data.

The definition file concept allows the handling of many catalogues by the SCF applications.

Catalogue Data Fields

SCF defines a number of fields that may be contained in any celestial cube catalogue. Right ascension and declination are compulsory fields as is a magnitude representation. Other fields such as colour, spectral class, constellation, reference numbers, duplicity and variability are options and are included at user request.

Catalogue Types

A 4 level hierarchy of star catalogues is contained within the Star Catalogue Facility:

- Catalogues obtained from external sources
  The source catalogues considered as input to SCF include: Hipparcos catalogue, Tycho catalogue, PPM catalogues and the Hubble Guide Star Catalogue (HGSC).
  Source catalogues are generally ASCII files with a fixed record format. The exception is the HGSC, which is organised as a series of FITS tables.
  In order to support the requirements for the XMM optical monitor the SCF design has been extended to also include access to the USNO catalogue.
- Master catalogue
  The master catalogue of stars will contain all the stars within the Star Catalogue Facility and will be derived from one or more source catalogues. Likewise a master catalogue of extended celestial objects is constructed.

- Mission Catalogues
  These will be derived from the master catalogue and will contain a subset of the master catalogue appropriate for the particular mission and may contain mission specific fields
- Run catalogues
  Run catalogues will contain a subset of the mission catalogue valid for a limited period of time and/or containing a limited sky area.

Transition Between Catalogue Types

The transitions between catalogue types are shown in Fig. 1.

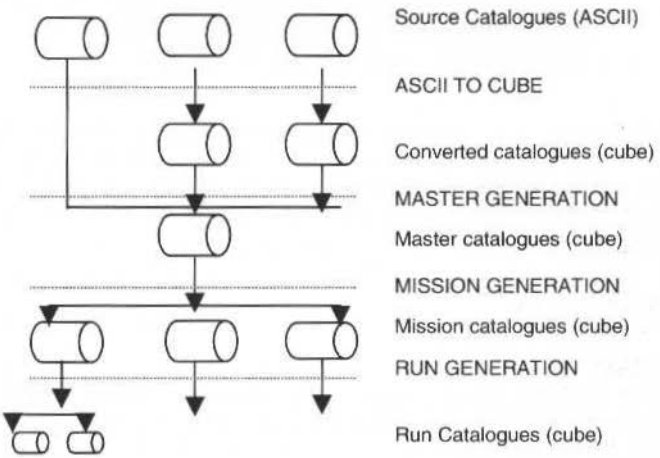


Fig. 1 Transition between catalogue types

ASCII source to Celestial Cube Format

The first step is to convert a source catalogue to the celestial cube format. As with catalogues in celestial cube format, a catalogue definition file, containing standard data such as reference system, epoch and equinox, is constructed for each source catalogue. For ASCII sources the definition file lists the data fields available by using keywords to define data types and for each data type available the units, start and end byte in the ASCII reference and default values must be listed. The standard definition file format allows reading of source catalogues through a single application.

Master Catalogue Generation

Master catalogue generation combines the cube versions of source catalogues. The sources are added one at a time, starting with the catalogues believed to be the most accurate. As each new source added SCF identifies which objects from the source catalogue are already included in the master catalogue and where appropriate the data record for such objects is expanded for each object to include new fields. All objects from the source catalogue not already included in the master catalogue are added as new records.

Mission Catalogue Generation

A mission catalogue comprises a subset of the master catalogue. At this stage additional mission specific or derived parameters may be added:

- instrumental magnitudes based either on a look-up table of instrumental magnitude offsets from visual as a function of spectral class;
- new fields constructed as numerical combinations of any other fields;
- classifications (e.g. guide star classifications) based on the logical combination of expression for field values or proximity to neighbouring objects.
- An example of guide star classification possible using SCF is given below:

One classification value could be given to stars brighter than magnitude 6 with no neighbouring stars or extended objects within 200 arcseconds, if this condition is not satisfied a second classification value

could be given to stars brighter than magnitude 7 with no neighbours within 300 arcseconds and so on. Such a guide star classification is typically used in observatory missions such as XMM, where the accuracy of a scientific observation is limited by the accuracy of tracking on a guide star.

The subset of stars to be taken from the master catalogue to include in a mission catalogue is defined similarly to guide star classifications, typically stars in a particular magnitude range are included.

A second method of constructing a celestial cube mission catalogue is allowed for - this is when a mission specific source catalogue is provided complete with mission specific parameters. In this case the cube mission catalogue is constructed directly from the ASCII source using the standard SCF application.

Run Catalogue Generation

A further reduction of stars may be made to include those valid for a particular epoch, by limiting to a specific region of the celestial sphere or ignoring stars close to solar system objects (SSOs) at a particular time. Typically a run catalogue may contain those stars expected to be viewed during a single orbit of an earth observation satellite such as ENVISAT.

Cube to ASCII

A further transition is possible from any cube catalogue to an ASCII file. The ASCII file may contain all or a subset of the source stars defined in an analogous manner to the inclusion of stars in a mission catalogue. The user may select all or some of the fields for output and define the field length and units.

In fact the ASCII file is produced with a corresponding catalogue definition file and may be further handled by SCF as a catalogue.

Other SCF Utilities

Catalogue Verification

Catalogue verification compares any two celestial cube catalogues, identifying common objects and identifying inconsistencies between the two catalogues. It is typically used to compare an imported mission catalogue with a master catalogue. Tables comparing the positions and magnitude of common stars are prepared and differences larger than predefined tolerances identified.

Statistics

SCF allows the generation of statistics based on the values of one or more fields within the catalogue. The specification for generating a statistic follows the format for generation of mission specific parameters. Any generated statistic may be plotted as a density map.

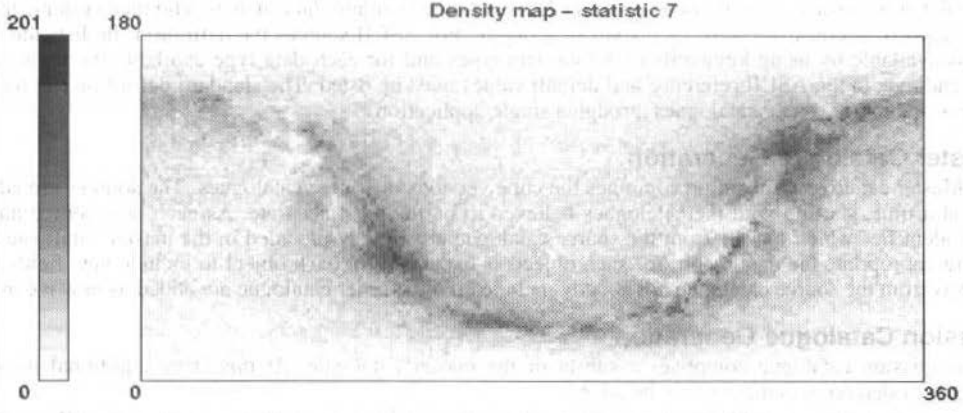


Fig. 2 SCF Density Map

Figure 2 shows a density map for stars between magnitude 10 and 12 from a master catalogue built from the Hipparcos and Tycho catalogues. The galactic plane is clearly visible.

Groups of statistics may be displayed as a histogram. Figure 3 shows a histogram generated from the same catalogue indicating stars in various magnitude ranges

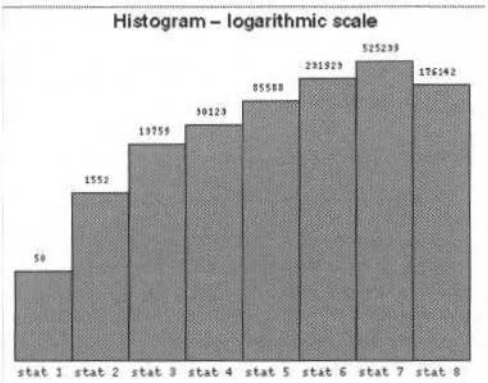


Fig. 3 SCF Histogram

Graphics

The SCF graphics tool allows the user to plot a field of view containing stars, extended celestial objects and solar system objects, taking its input from any celestial cube star catalogue, any celestial cube catalogue of extended objects and a master set of SSO ephemeris. Instrument fields of view may also be plotted..

A suite of subroutines is provided to allow a user to build their own plots tailoring the layout to the needs of specific missions.

A sample plot is shown in Fig. 4. In addition to stars, extended objects and solar system objects in a star tracker field of view, the plot shows objects retrieved from telemetry, star tracker blemish an instrument field of view within the star tracker field of view and a raster pattern. The guide star is also identified.

The graphics allow zoom features, the measurement of distance between any two points on the graph and the pop up display of object data on any selected object, an example of which is shown in Fig. 4.

Generic Attitude Software (GAS)

The generic attitude support (GAS) software provides a library of subroutines for use in attitude applications, which make use of star data. The GAS subroutines are derived from applications developed to support earlier ESA missions, such as HIPPARCOS and ISO and are developed to be fully consistent with the ORATOS infrastructure.

For spacecraft where the attitude evolves slowly due to controlled rotation rates a star transit time predictor is developed. Spacecraft of this type include Hipparcos and Envisat. The software is also applicable to 3-axis stabilised spacecraft following closed/open slew between targets (e.g. XMM and Integral)

For three axis stabilised spacecraft the GAS software provides subroutines for star pattern matching, attitude determination and guide star selection. Spacecraft of this type include ISO, XMM and Integral.

The GAS subroutines extend the functionality of the Star Catalogue Facility to provide a reusable library of routines for the development of attitude monitoring and command generation systems to support ESA missions or specific phases of ESA missions

Conclusions

The first Star Catalogue Facility developed for ESOC was not used for the Exosat mission because of the parallel development of the Exosat software. The next mission, Hipparcos, benefited from the availability of the INCA catalogue for the support of that mission and therefore had no need for a generic facility. The facility was ported from ICL 4/72,to Honeywell Bull,to IBM and finally archived on SUN.

The design for the new Star Catalogue Facility therefore attempted to account for the requirements of anticipated future mission and at the same time built on the experience gained in the operations of the earlier missions. The design was such that the facility can be included in the context of the ORATOS infrastructure for ESOC Flight Dynamics.

The result of this approach was the development of a facility which is both usable by ESOC missions and which is actually being employed to support the preparation of the ground control systems.

Current uses of the star catalogue facility include in support of ESA missions include:

- XMM/ Integralmission catalogue
- Special catalogue for planning observations of the XMM Optical Monitor instrument
- Validation of the Envisat star catalogue
- XMM/Integral Flight Dynamics software
- Rosetta requirements analysis

Acknowledgements

The author acknowledges the assistance provided in the course of SCF development by Mr. Xavier Marc of ESOC and the help of Mr. Alain Schütz of ESOC in preparing the historical perspective and reviewing the text of this paper.

Mark Tuttlebee of SSL carried out the development of the generic attitude software library.

The work reported in this paper was carried out under ESOC contract 8265/89/D/IM

References

RD1

A. Schütz A Fast method to Retrieve Data from a large Star Catalogue File. Automated Data Retrieval in Astronomy 1982, D.Reidel Publishing Company

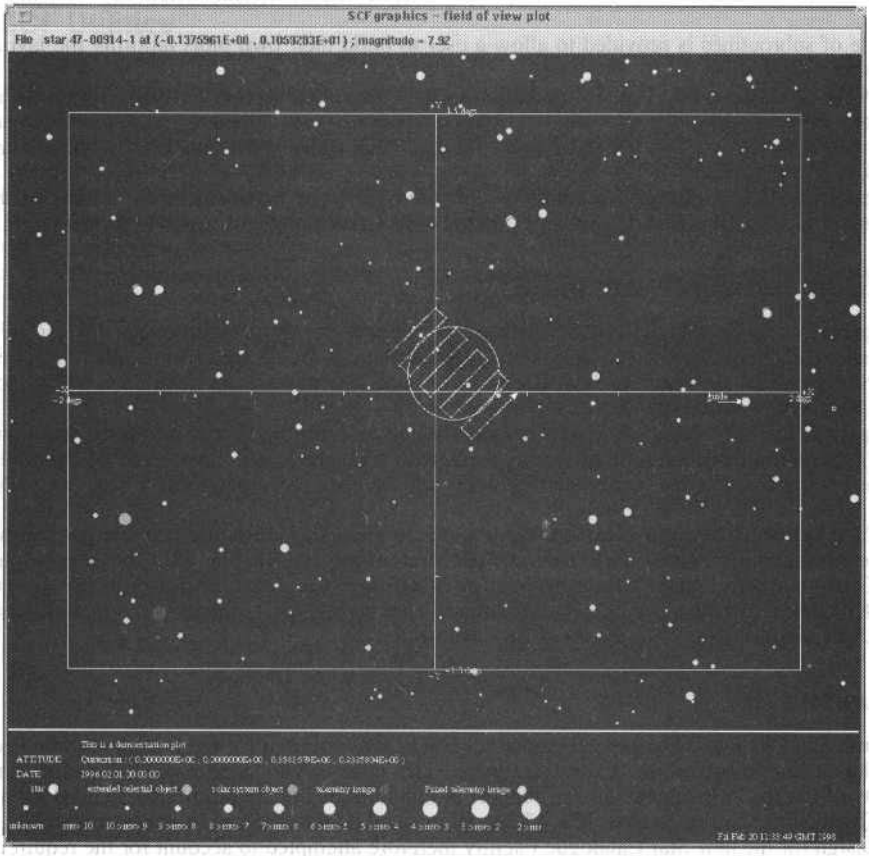
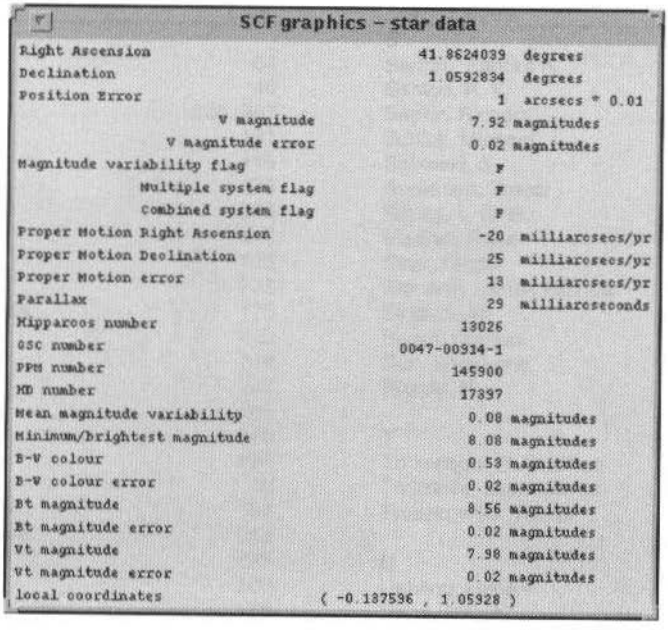


Fig. 4 Typical SCF graphics display with pop-up



SCF graphics – star data

Right Ascension	41.8624039	degrees
Declination	1.0592834	degrees
Position Error	1	arcsecs * 0.01
V magnitude	7.92	magnitudes
V magnitude error	0.02	magnitudes
Magnitude variability flag	F	
Multiple system flag	F	
Combined system flag	F	
Proper Motion Right Ascension	-20	milliarcsecs/yr
Proper Motion Declination	25	milliarcsecs/yr
Proper Motion error	13	milliarcsecs/yr
Parallax	29	milliarcseconds
Hipparcos number	13026	
GSC number	0047-00914-1	
PPM number	145900	
HD number	17387	
Mean magnitude variability	0.08	magnitudes
Minimum/brightest magnitude	8.08	magnitudes
B-V colour	0.53	magnitudes
B-V colour error	0.02	magnitudes
Bt magnitude	8.56	magnitudes
Bt magnitude error	0.02	magnitudes
Vt magnitude	7.98	magnitudes
Vt magnitude error	0.02	magnitudes
local coordinates	(-0.187596 , 1.05928)	

RBCM - J. of the Braz. Soc. Mechanical Sciences
Vol. XXI – Special Issue - 1999 - pp. 514-515

ISSN 0100-7386
Printed in Brazil

Author Index

(Continuation)

Posters

(Continuation)

(Continued)

ISSN 0100 - 7386

JOURNAL OF THE BRAZILIAN SOCIETY OF MECHANICAL SCIENCES
REVISTA BRASILEIRA DE CIÊNCIAS MECÂNICAS

VOL. XXI - SPECIAL ISSUE - 1999

SCD II - 2nd International Symposium on Spacecraft Ground Control and Data Systems
Foz do Iguaçu, PR, Brazil, February 8-12, 1999

(Continued on Inside Back Cover)