A Ground Control System for CBERS 3 and 4 Satellites

Paulo E. Cardoso, Joaquim P. Barreto, Kléber M. Dobrowolski, Luciana S. Cardoso, and Leandro T. Hoffmann
National Institute for Space Research – INPE, Sao Jose dos Campos, SP, 12227-010, Brazil

Email: {paulinho, joaquim, kleber, luciana, hoffmann}@dss.inpe.br

The ground control of the third and fourth China-Brazil Earth Resources Satellites (CBERS 3 and 4) will be carried out by a new system under development at INPE. This system will include new technologies to reduce costs and development time of future ground system projects, through shared or adaptable software. Taking advantage of the experience gained in earlier ground control systems, the entities related to satellite operations activities have been modeled as metadata. This modeling approach will increase the systems´s reusability, reducing the efforts required to make changes. Whenever modeling costs of entities via metadata have been become unfeasible, their programming interfaces have been defined by Design Patterns in order to facilitate future changes. Furthermore, the telecommand and telemetry subsystems architecture includes a processing kernel, which can be shared with EGSE and satellite simulation software. This architecture also foresees the inclusion of CCSDS communication link protocols in other INPE satellite missions. In the future, this system will be upgraded to include Artificial Intelligence (AI) techniques, like Planning, to prepare flight operation plans for the routine phase of the missions. The engineers in charge of planning the satellite operations will use tools developed with Dynamic Object Model technology to define the operations activities. Through the high-level script language supplied by these tools, the engineers will be able to define and/or change the knowledge database of operations activities, without requiring specific software development for each satellite. In addition, these tools will make it easier to define planning goals and to edit existing planning to deal with non-routine operations activities. The operations plan will have actions that are automatically executed by the system, as well as actions that are manually executed by human operators. This paper presents the modeling of the operations entities to increase the satellite control system´s reusability and provide software that may be used by the control center, testing (EGSE), and simulation systems. This paper also shows how the Dynamic Object Model and AI technologies will be added to the system in order to automate the control center operations, thereby reducing operations cost.

I. Introduction

At the moment, INPE is operating three satellites: SCD1, SCD2 and CBERS-2. The Data Collecting Satellites (SCD1 and SCD2) are part of the Brazilian Complete Space Mission (MECB) and have as their objective to receive data transmitted by data collecting stations spread over Brazilian territory and retransmit it to the mission center in Cachoeira Paulista where the data is processed and made available to the community. The SCD1 satellite has been in operation since 1993 and the SCD2 since 1998. The CBERS-2 satellite is the second remote sensing satellite developed with China as part of the China Brazil Earth Resource Satellites program. The first CBERS satellite was launched in 1999 and was controlled by INPE and China up to the end of its life (2003). The agreement with China was renewed for the development of three more remote sensing satellites (CBERS-2B, CBERS-3 and CBERS-4). Therefore, the Tracking Control Center (CRC) needs to be prepared to control a larger number of satellites. Since the CBERS-2B satellite is quite similar to the CBERS-1 and CBERS-2 satellites, the control system in use will be improved to meet specific requirements of the CBERS-2B satellite. However, new control system is being developed for CBERS-3 and CBERS-4 satellites.

The development of the Satellite Control Systems is under the responsibility of the Ground System Development Division (DSS) at INPE. Since the beginning of its activities, DSS has aimed to develop systems whose architecture permitted maximum reuse for operating other satellites. The DSS team has already developed three generations of

Satellite Control Systems and, in each system, new technologies have been used in order to achieve the reusability goals. The first system developed by DSS was named SICS (Satellite Control System) for the purpose of controlling MECB satellites. To reach the reusability goals, SICS used configuration files[1]. Afterwards, the Telemetry and Telecommand System (TMTC) was developed, using an Object-Oriented Programming architecture and relational databases. The main purpose of TMTC was to meet CBERS requirements and replace SICS in controlling the MECB satellites. While SICS used VAX/Alpha machines, which had high maintenance costs, the TMTC system used microcomputers[2, 3, 4]. The third generation of control systems was developed to comply with the operations requirements of scientific micro-satellites. The advances in this third generation were a higher database generalization and a telemetry and telecommand framework that could be more easily customized for different missions. This made it possible to fulfill the operations requirements of three different satellite families (MECB, CBERS, micro-satellites) with a reduced number of changes. The project introduced the use of *Design Patterns*[5] to increase the framework´s flexibility.

With each new mission the development team has increased system flexibility, allowing greater reusability with fewer modifications. The fourth generation now under development for the CBERS 3 and 4 satellites, named SATCS (SATellite Control System) will maximize reuse and comply with two new requirements:

- create applications that can be easily customized to be used in the Ground Segment Operations, as well as in the Assembly, Integration and Testing (AIT) and;
- automate the Ground Segment Operations.

This paper presents the architecture of the SATCS system which is based on a layered architecture: Kernel, Application and Automation. The first layer, Kernel, is described in section II. Besides a general overview of its software modules, the Meta Instance module will be presented with more details. This module allows modeling the entities concerning the satellite's operations activities as metadata. Section III describes the Application layer which contains the satellite's control applications. The design of this layer was oriented to reach higher system adaptation flexibility for new missions and to build applications that can be used either in operations or AIT phases. The top layer, Automation, provides the automation of some satellite operations activities at INPE. The design of this layer, which includes Artificial Intelligence[6] and Dynamic Object Models[7] (DOM) technologies, is presented in section IV. Section V presents the conclusion of this work.

II. Kernel Layer

The purpose of the Kernel layer is to provide the Applications Layer with a high level API with functions for data communication, timer events, event log, date time conversion and operations, modeling structures using metadata and services to access the configuration and historic databases used by the satellite control applications.

Figure 1 presents the Kernel layer, its three sub-systems and its two databases: the Operations Database Management Sub-System (ODM), the Supporting Sub-System (SPT), the Archiving and Retrieving Data Sub-System (A&R), the Operations Database (OPDB) and the Historical Database (HISDB) . Together these sub-systems and databases provide a basic group of services that are used by the upper layers. Both ODM and A&R sub-systems directly access persistent data stored in databases, providing services to handle these data. SPT is a support sub-system, responsible for implementing several comprehensive services. All the kernel services are accessed through standard interfaces.

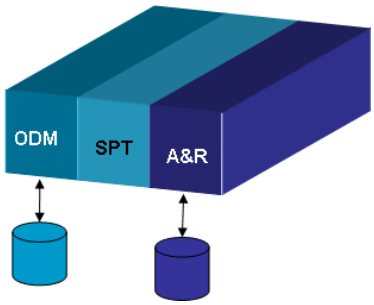

Figure 1. Kernel Layer

ODM manages an operations database which represents the satellite control entities through metadata as much as possible. The metadata is also used to make it easier to reconfigure the SATCS software applications. Once the control applications work with metadata, it is more feasible to reuse them to control different satellite missions and ground stations.

A&R archives and retrieves the mission´s satellite and ground data history such as logbooks, telemetry packets, remote monitoring messages, doppler measurements, operations plans and pass planning information.

By using the SPT sub-system, it is possible to isolate some operating system implementation details in the development of the Application Layer. Some of the SPT features include:

- **Communication**: supports synchronous and asynchronous communication services over TCP/IP protocols, using sockets;
- **Configuration Param**: retrieves user configuration parameters stored in the operations database;

- **Event Message**: stores log messages generated during the execution of the SATCS applications;
- **Event Visual**: provides services to visualize log messages generated by SATCS applications;
- **File Manager**: transfers and deletes files over LAN or Internet, using FTP, SSH or NetBIOS protocols;
- **Special Types:** contains classes to handle special types of data (for instance: date-time, and arrays of bits);
- **Timer**: schedules services to be executed in a defined time;
- **User Interface**: implements general user interface services (for instance: date-time input masks) and;
- **Meta-Instance**: Processes metadata information stored in the operations database (message coding and decoding).

From the point of view of reusability, the most important feature in this layer is the Meta Instance module, which allows for defining a generic data structure by using metadata. The other modules already were part of the previous versions of the satellite control system and in this version were just improved. Since the Meta Instance module is one of the main characteristic of the new version, the next item describes it in more detail.

A. The Meta-Instance Module

The Meta Instance module moves the data structure definition from code to the operations system database. This allows changes in data structure without changing the code. This module is composed of a packet of C++ classes, which uses entities defined in the operations database to create their objects during runtime.

It is possible to define a data structure as a hierarchical tree of data fields in the database. One data field can be either a sub-structure or a parameter (atomic entity). Structures can be multiplexed, meaning their composition varies dynamically in runtime, according to some other parameter values. The parameters can be associated to a transfer function that converts raw values to engineering values (and vice versa) and also to monitoring functions. Both monitoring and transfer functions can be conditioned to other parameter values, which permit the representation of several behaviors according to the states of the satellite subsystems. Special transfer functions may also be defined through a DLL. Once data structures are defined, it is possible to create instances of them and assign values to each one of their parameters.

The class interface of this module makes a group of services available to the client applications. These services enable user clients to: obtain/define parameters values; generate a string of bits that represents a data structure; and obtain parameters values from a string of bits that represents a data structure.

The metadata concept has been used in the development of this module. Metadata are structured, encoded data that describe characteristics of information to aid in the identification, discovery, assessment, and management of the described entities. Metadata modeling in the SATCS system permits a unique representation for different kinds of data such as telemetry messages, telecommand messages, communication protocol layers, or other data that can be represented as a hierarchical data tree.

All the metadata are stored in the database through the following information categories: MetaFields, Instances and Applications. The MetaFields category is the first one to be created in the database. A data message can be described as a hierarchical data tree structure. Figure 2 shows an example of the hierarchical tree structure of a data message. Each level is composed of one or more data structures (branches). These data structures are composed of other data structures (new branches) or parameters (leaves). Using this approach, it is possible to model several kinds of data messages in the database. It does not matter how big or complex the data message is.

The representation of a tree structure in an object-oriented software may be made by the Composite Design pattern. This pattern composes objects into tree structures to represent part-whole hierarchies so that individual objects and compositions of objects can be treated uniformly[5]. The translation of the Composite pattern to a relational database model was done using three entities (MetaField, MetaStructure and MetaParameter) and one relationship (Composed of), as seen in Figure 3.

The entity MetaField is abstract and is specialized through the MetaStructure and MetaParameter entities. Every MetaField must be either a structure or a parameter. The MetaStructure entities are branches, while MetaParameter entities are leaves. The composition of a hierarchical data tree is defined through the relationship “Composed of”. Each structure can contain other structures or parameters.

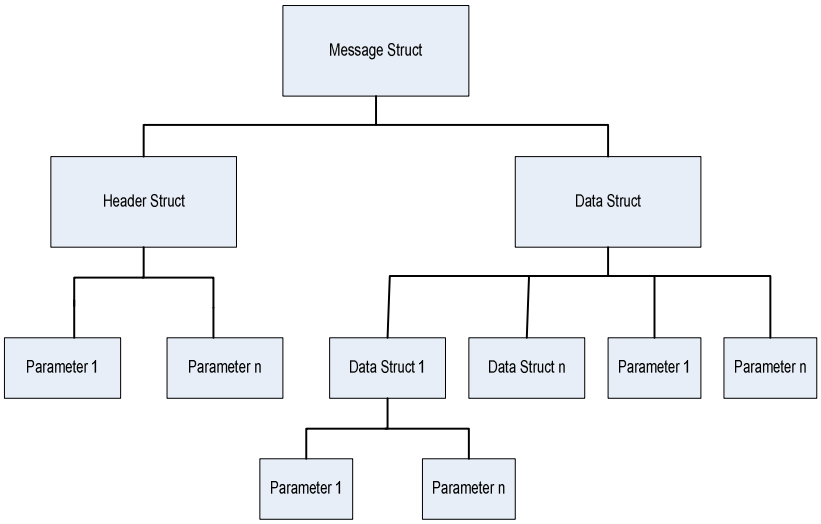

Figure 2. Data message hierarchical tree structure

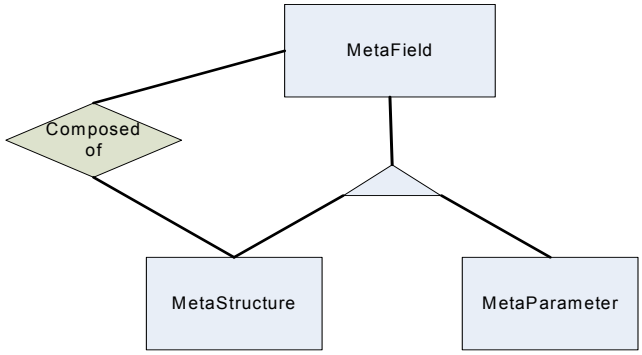

Figure 3. Composite Pattern modeled to a Relational Database

After defining the MetaFields, the next step is to create the Instances category. As in Object Oriented Programming, the MetaFields category may be stated as classes and the Instance category may be classified as the objects of these classes. For example, to model a telemetry message, it is necessary to create its instance in the database and make the relationship between this instance and the MetaField entity that represents the root structure of the telemetry message. To model the Telecommand messages, one instance must be created for each single telecommand and then the relationship must be made between each instance and the MetaField (main structure) that represents the telecommand message.

Finally, after the creation of the MetaFields and MetaInstances categories, the last step is to create the Applications category. In this category the group of instances which are used through an application are defined. When an application is started, only its instances are recovered from the database.

Once all this information is stored in the operations database, the satellite control applications can use the Meta-Instance module to decode or code data messages, regardless of their contents.

III. Application Layer

The Application layer concentrates all the software applications necessary for controlling the satellite, like Telemetry and Telecommand (TMTC), Ranging (RAN), Range Rate (RR), Flight dynamics (FD) and Ground Station Monitoring and Control (RCM).

A template architecture was defined for developing the SATCS applications, as shown in Figure 4. The architecture is composed of four modules. The User Interface module implements all the application user requirements. The Communication module implements all the requirements for the interface communication with other software systems or equipment, such as interface communication with ground stations and satellites. The

Process module implements all the application functional requirements, such as data message coding, data message decoding and data monitoring. The history module concentrates all the data exchange functions (storing/retrieving) related to the history database. The interfaces among these modules are designed, using the Facade design pattern to let one module deliver services to the other ones, without exposing its implementation details. All these modules make use of the Kernel layer and are designed using other design patterns, such as Builder, Composite, Factory, Strategy and States[5] whenever possible. These patterns provide a higher independence among the modules thereby creating flexibility to change specific points without affecting the entire application.

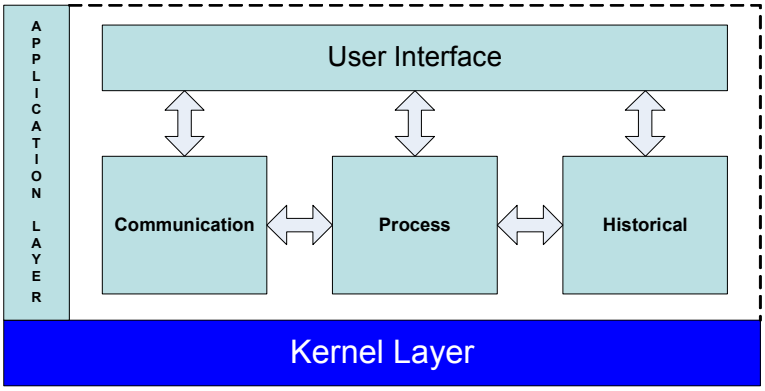

Figure 4. Application Template Architecture

This development philosophy benefits the satellite control system in three ways:

a- By using the Kernel layer, it is possible to reduce drastically, or, in some cases, even to eliminate the need to adapt the application codes to comply with the requirements of new satellites. A large number of requirements can be implemented by configuring the satellite operations database;

b- By using Design Patterns, its is easier to include new requirements, since the code is designed to accept changes, thus saving time and reducing future developments costs; and

c- The module design also makes it easier to change a specific part of the problem domain without modifying other modules, such as changing the ground-ground or the ground-space communication protocols, because the change would be concentrated in a specific module (in this case, the communication module).

A version of SATCS can be configured for each satellite controlled by INPE (CBERS-2, SCD-1 and SCD-2) as well as for the CBERS-2B, CBERS-3 and CBERS-4 satellites that are under development. The SATCS can also be easily adapted and configured to control INPE scientific satellites (MIRAX and EQUARS) that will use the CCSDS protocol for ground-space communication. Most of the operations requirements can be fulfilled by configuring the operations database with the specific parameters of each satellite, as shown in Figure 5.

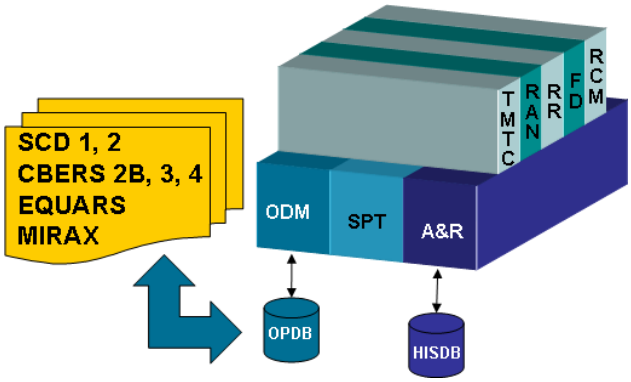

Figure 5. SATCS Configured for INPE Satellites

Another purpose of the SATCS Application layer is to create a Telemetry and Telecommand Processing kernel (TMTCPROC) that can be used in satellite control operations and also during the AIT phase.

This will permit:

- End to end validation of the complete ground segment and space systems;
- Re-use of space segment operations data between AIT and mission operations; and
- Commonality between EGSE and the Mission Control Center.

Figure 6 shows the reuse of TMTCPROC for several applications. In addition to the in-flight operations systems and AIT, the services of this kernel can also be used in the satellite simulator project and for testing the on-board computer during its development. The onboard computer development and testing team will customize the TMTCPROC to monitor and control the satellite on-board equipment during the testing phase by inserting new equipment data in the operations database. This will replace several tests programs which currently perform tests, requiring a separate program for each piece of equipment. In the new architecture, to test equipment it will only be

necessary to insert its data in the operations database. The same on-board computer test application can be used to test all the equipment.

In the AIT phase, this kernel will be used inside EGSE to control and to monitor the satellite during testing, thus eliminating the need to develop two nearly identical systems, one for AIT and another for ground segment operations.

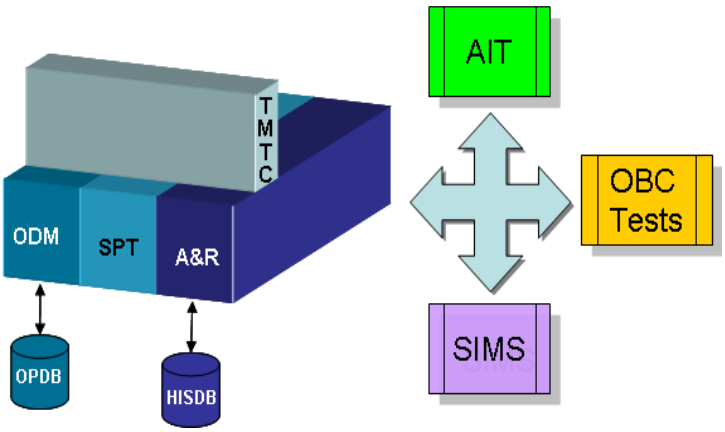

Figure 6. New Application based on TMTCPROC

The TMTCPROC can be customized for the Satellite Simulator (SIMS) to:

- generate the telemetries data structure frames that will be sent to the Satellite Control Center (through fixed values on the database or dynamic values created by the simulation models);
- decode the telecommands received from the Satellite Control Center; and
- code/decode the data structure of the protocols used for communication between the Satellite Control Center and the ground station simulator.

It is important to emphasize that along with the TMTCPROC functions, all the Kernel layer services will be available to these users reducing the number of code lines required to be developed by several INPE´s groups. The system flexibility creates opportunities for new ways of using it, in satellite control or for other INPE control and monitoring systems.

IV. Automation Layer

Most of the activities in satellite control operations are routine tasks that can be automated somehow in order to decrease the amount of work that engineers and satellite controllers must perform manually. Therefore, one more layer will be created to automate part of the tasks in the routine operations phase. To account for the operators´ understandable reticence in transferring operations activities to a totally automated system, a semi-automated one is under development. The main purpose is to permit the control of additional satellites in operation without having to increase ground personnel to operate them, thus reducing overall operating costs. Automation can also reduce human errors during the execution of well-known repetitive activities, which will lead to a more reliable system. The operations activities will be automated, but an operator can take control at any time. It is expected that once the simpler repetitive activities have been executed successfully by the automated system, the operators will feel confident to delegate more complex activities to this system.

The proposal is to develop two more subsystems: Planning (PLN) and Activities Scheduling (SCH). These subsystems will also use the Kernel layer. The SCH subsystem will have an interface with all the Application layer subsystems to activate their services and monitor their states. Figure 7 shows the SATCS architecture with the two new subsystems.

The PLN subsystem will work in two stages of operations planning at the Satellite Control Center: multi and mono satellite.

In the multi-satellite stage, one module of the subsystem will be responsible for allocating the available ground stations to the several satellites in operations. This allocation will consider the simultaneous pass conflicts such as: more than one satellite passes over the same ground station, one

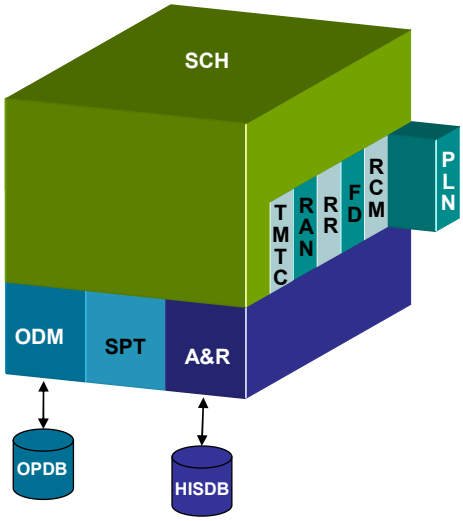

Figure 7. Complete Architecture of SATCS

satellite passes over two or more ground stations at the same time and the minimal time interval between the end of one satellite pass and the beginning of another satellite pass over the same ground station. As a result, the Pass Visibility Prediction (PVP) will be generated for each pair (satellite, ground station) containing only the complete or partial passes that will be actually tracked.

In the mono satellite stage, another subsystem module using the AI temporal planning technology[8] will be responsible for generating the Flight Operations Plans (FOP) for each satellite. The module will, therefore, make use of a knowledge base with the behavior of the tracking domain of INPE's satellites and planning problem instances. The problem instances will define the initial state environmental of the satellite passes and the goals to be reached. Figure 8 illustrates the architecture of the PLN module which is responsible for this intelligent (AI) planning.

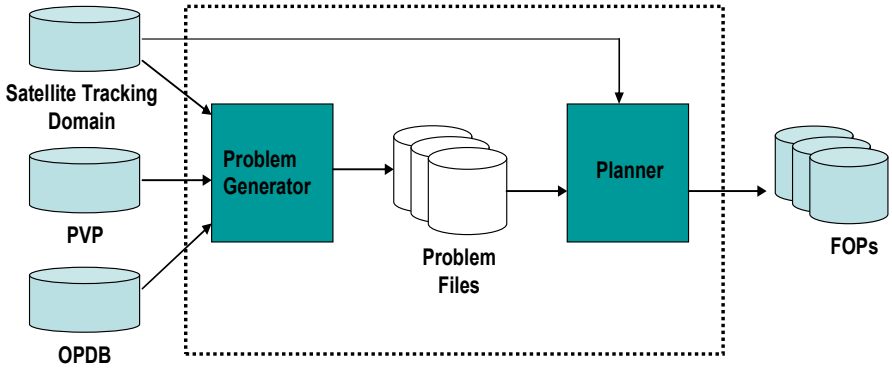

Figure 8. The Intelligent Planning Module

In addition to the PVP files and the knowledge base, the ground stations and satellite operations characteristics will also be used by PLN to create all the problem instances automatically. Afterwards, PLN will trigger the planner responsible for generating the FOP files.

The planning characteristics for ground stations and satellites will be stored in the operations database where they will be able to be modified by the engineers responsible for controlling the satellites. This provides the necessary flexibility to modify the satellite operations scenarios and insert new ground stations for tracking the satellites without updating the subsystem code. Each FOP will be stored in the A&R subsystem database, providing historical data of the operations activities planned for the mission.

The SCH subsystem will be responsible for preparing a schedule of activities, using the plans generated by the PLN subsystem to execute each activity at a pre-defined time and to control its execution. The activities will be executed automatically by an application of SATCS or, in the case of critical activities, they might be assigned to an operator.

The control mechanism of the execution of each activity will be defined in the operations database and will be used by SCH to make decisions after the execution. When SCH detects any problem in the execution of an activity that has been programmed to be executed automatically it can:

- Run a recovery procedure previously defined; or
- Change the execution of all activities to manual control, in the absence of a recovery procedure or in the case of errors during the execution of a recovery procedure.

The status of the automatic executions as well as the reports about manual executions of the activities will be recorded in the A&R subsystem in order to update the mission history.

A sharing mechanism of the object states of each application will allow the SCH subsystem to control the execution of a sequence of activities. This sharing mechanism will consider the possible distribution of the applications in different machines. The SCH could, for example, check the state of a shared object to decide whether it should trigger an activity associated to a second application.

Dynamic Objects[9] (DOM) technology will be used to create a high-level language for the satellite´s operations domain. The purpose is to make it easier to define procedures of activities. DOM will also allow operations engineers to configure PLN and SCH subsystems directly for different missions or satellites. Various data will be defined as metadata in the ODM subsystem: the state and the behavior of objects associated with satellites operations; objects of basic types (integer, string, Boolean, float); and control methods (conditioned and/or iterative executions). Engineers will have an editing tool that will naturally combine message exchanges among existing objects to create scripts.

The design of the PLN and SCH subsystems will comprise those parts of the code which are most likely to require changes to meet the specific needs of each satellite or mission. The design of each of these parts will consider the access to scripts created according to the DOM technology. The operations engineers will be able to adapt the system through the creation of suitable scripts for each situation.

V. Conclusion

The SATCS is a solution presented by DSS and goes beyond answering the operations requirements of CBERS-3 and CBERS-4 satellites. It will permit the fourth generation to be more easily configurable for other missions, reducing code changes and introducing automation in ground operations, a feature which was unavailable in previous versions.

The SATCS layered development, together with the use of metadata concepts and Design Patterns in the software design permit the creation of a system that is easily configurable for several missions since most of the possible changes are concentrated in the database configuration. This configuration flexibility occurs especially because of the Meta Instance module which moves the definition of data structures from code to the system´s operations database.

The design approach of the system allows its use in the AIT phase and satellite simulators as well as for satellite operations. A huge economy can be obtained in the total cost of mission development and operations by reducing the need for new code and by facilitating the integration of ground and space segments.

Finding automated alternatives for the satellite operations activities at INPE is an important issue in order to maintain satisfactory performance of these activities despite the scarcity of financial resources. The insertion of PLN and SCH subsystems in SATCS architecture satisfies the requirement of automation of the satellite operations during the routine phase. The AI Planning technique is the solution presented to automate the generation of FOP files. And the use of DOM technology makes it possible to transfer the responsibility of modifying system behavior to the operations engineers.

The automation solution adopted allows transferring control to the manual mode whenever necessary. Current automation programs should lead to more AI software in the future.

References

[1]Yamaguti, W., Vieira, A. E. C., Oliveira, J. L., Cardoso, P. E., Costa, P. O., "Satellite Control System Nucleus for the Brazilian Complete Space Mission," *International Symposium on Ground Data Systems for Spacecraft Control*, SEE N91, Darmstadt, 1990, pp.87-90

[2]Cardoso, P. E., Gonçalves, L. S. C., Ferreira, M. G. V., Ambrosio, A. M., "Brazilian Experiences in Upgrading a Satellite Control System," *IV International Symposium on Space Mission Operations and Ground Data Systems*, Munich, German, 1996, URL: http://www.esoc.esa.de/external/mso/SpaceOps/2_31/2_31.htm [cited 15 March 2005].

[3]Cardoso , P. E., Gonçalves, L. S. C., Ambrosio, A. M., "Lessons Learned in Adopting PCs at the Brazilian Satellite Control Center," *V International Symposium on Space Mission Operations and Ground Data Systems*, SPACEOPS98, Tokyo, Japan, 1998, URL: http://track.sfo.jaxa.jp/spaceops98/paper98/track5/5c007.pdf>, [cited 15 March 2005].

[4]Gonçalves, L. S. C., Cardoso, P. E., Ambrosio, A. M., "Satellite Control Center: Solution for an Adaptable System," *IV International Symposium of Small Satellites Systems and Services*, SSSS, TL795.4.S27, Antibes, France, 1998

[5]Gamma, E., Helm, R., Johnson, R., Vlissides, J., *Design Patterns: Elements of Reusable Object-Oriented Software*, 1st ed., Addison-Wesley, Massachusetts, 1995, pp. 395.

[6]Weld, D. S., "Recent Advances in AI Planning," *AI Magazine*, Vol. 20, No. 2, 1999, pp. 93-123.

[7]Yoder, J. W., Balaguer, F., Johnson, R., "Architecture and Design of Adaptive Object-Models," *ACM Sigplan Notices*, Vol. 36, No. 12, 2001, pp. 50-60.

[8]Cardoso, L. Ferreira, M. Orlando, V. Biancho, A., "Aplicação da Tecnologia de Agentes de Planejamento em Operações de Satélites" *VII Simpósio Brasileiro de Automação Inteligente*, São Luís, Maranhão, 2005.

[9]Cardoso, P. E., "Uma Nova Arquitetura para a Representação das Regras de Negócio em Modelos de Objetos Dinâmicos", Dissertation, Computação Aplicada, INPE-13044-TDI/1020, INPE, São José dos Campos, SP, 2005.