

AN OBJECT-ORIENTED MISSION CONTROL SYSTEM DEVELOPMENT: A PROGRESS REPORT ON SCOS-II AND ITS CLIENT PROJECTS

M Jones¹, A Baldi¹, P Maigné¹, P Howard², R Melvin³, W O'Mullane⁴, S Lynenskjold⁵

1. European Space Operations Centre (ESOC)

Robert-Bosch-Str. 5, 64293 Darmstadt, Germany, FAX No.: +49 61 51 90 30 10

E-Mail: MJONES@ESOC.ESA.DE, ABALDI@ESOC.ESA.DE, PMAIGNE@ESOC.ESA.DE

2. Science Systems Limited, 23 Clothier Road, Brislington, Bristol, BS4 5PS, UK, FAX No.: +44 117 972 1846

3. VEGA, 2 Falcon Way, Shire Park, Welwyn Garden City, Herts AL7 1TW, UK, FAX No.: +44 1707 39 39 09,

E-Mail: RMELVIN@ESOC.ESA.DE

4. CARA, Palmerston House, Fenian, Street, Dublin 2, Ireland, FAX No. +49 61 51 90 34 14,

E-Mail: WOMULLANE@ESOC.ESA.DE

5. Computer Resources International A/S (CRI), Bregnerødvej 144, 3460 Birkerød, Denmark, FAX No. +45 45 94 94 01, E-Mail: STEEN@ACM.ORG

Abstract: This paper reports on the development of a new infrastructure SCOS-II (Spacecraft Operations System-II) for building Mission Control Systems. The development started in 1992 and at the time of writing (June 1996) has reached release 1.2. It is a very ambitious project, involving embarking several new technologies as compared with its predecessor system: UNIX platforms (replacing the previous VAX / VMS platforms), a distributed system architecture (replacing the former centralised one) and object-oriented development techniques, to name the main ones. In addition, SCOS-II aims to extend the number of functions supported.

Papers were presented on SCOS-II at SpaceOps 92 and SpaceOps 94, mainly concentrating on the concepts and early prototyping work. At this Symposium, we take a more retrospective look at the project, since deliveries have been made covering many basic functionalities and substantial work has been done on a number of "client" missions.

1. INTRODUCTION

This paper presents the status of the development of ESA's new mission control infrastructure, the second generation of the Spacecraft Control Operations System SCOS-II. The paper also discusses the evolution of the project, reflecting the effect of a certain amount of re-direction and change in scope that has taken place during the past two years or so. To this end we discuss the strategic aims and overall approach (Chapter 2). We then discuss (Chapter 3) the scope of the project envisaged at the time of the requirements analysis phase in 1992 - 3. Chapter 4 discusses the actual architecture and scope currently being implemented. Some key lessons learned are described in Chapter 5. Chapter 6 describes the status of the implementation for a number of missions.

2. OVERALL APPROACH OF SCOS-II

The approach adopted can be summarised as follows:

1. Object-Oriented technology is used to permit easier reuse of SCOS-II software - this is expected to reduce the amount of mission-specific software needed.
2. SCOS-II will provide all of the functionality of its predecessor, plus telecommand functions, on-board software maintenance, a tool to support flight operations plan generation (FOPGEN) and basic functions for processing of "historical" data in support of spacecraft performance analysis. Various advanced

functions were also envisaged in the requirements analysis phase (as discussed in Chapter 3). These are not included in the present development for reasons discussed in Chapters 3 - 4. An independent system acting as a communications front-end for ground stations is also provided (see Chap. 5).

3. To address vendor independence and performance the SCOS-II system runs on a Local Area Network of UNIX workstations. The use of UNIX, combined with careful use of Third-Party software and use of suitable C++ coding conventions also aims to allow vendor independence, that is, easy porting of the system to other UNIX workstations.

Another aim is to provide a modern maintainable system. The SCOS-IB infrastructure is written mainly in FORTRAN and it is becoming more difficult to find development and maintenance staff to work with such old technology.

3. ORIGINAL AIMS: ADVANCED FUNCTIONS

The SCOS-II project as originally conceived was very ambitious. For the first time for an infrastructure, an extensive effort was undertaken by the users (ESOC's Mission Operations Department) to write a comprehensive User Requirements Document (URD). This attempted to go beyond the scope of classical spacecraft control systems, and the work involved close interaction between the users and software developers. In particular the activities of writing user requirements, software requirements analysis and prototyping were done in close cooperation.

A number of new ideas came of this early work (1992 - 94),

- "system elements": a system element provides a high-level view of the unit which it represents: activities e.g. command procedures can be associated with it and a system element can contain subordinate system elements. This led to a number of related ideas such as element templates, navigation (at the user MMI) through system element hierarchies, procedure manipulation as part of the system element, and model-based control, etc. More detail can be found in Jones et al (1994) and Kaufeler et al (1994).
- Transmission route modelling, using the concept of system elements to model the end-to-end flows of data from spacecraft to control system, modelled as a network of nodes joined by "route segments" ..
- Advanced spacecraft database concepts, such as allowing database "parts" for specialists responsible for the subset of the database corresponding to e.g. one spacecraft subsystem.
- Operations language, which allows users who are not software engineers to specify spacecraft knowledge (e.g. derived parameters).
- Separation of user interface from the underlying functionality, allowing much more freedom to modify and extend user interfaces..

4. ACTUAL IMPLEMENTATION AND ARCHITECTURE

In early 1995, a detailed review of the project took place. Also, by this time, the main client missions of SCOS-II were clear. Results of the reflection resulting from this review and the assessment of real project needs led to the following conclusions:

1. The client missions predominantly required classical mission control functions.
2. Neither time nor available funding permitted provision of the advanced functionality described in Chapter 3.
3. Even without this advanced functionality, the scope of SCOS-II would still be considerably wider than

that of either two existing infrastructures.

4.1 ARCHITECTURAL OVERVIEW

SCOS-II is based on the client-server concept, with servers providing common services such as telemetry processing, data archiving and retrieval, command uplinking, etc. to a set of client workstations for the operational users. To provide “scaleability”, that is the capability to expand the network size without proportionally increasing the load on the servers, a telemetry data broadcast technique is used, as described in Baldi & Pace (1994).

The software architecture has some 15 subsystems. These are split into two categories:

- “Middleware” (History File Archiver / Cache, Utilities, Object Store, Mechanisms).
- Applications (the rest of the subsystems e.g. monitoring model and applications, TC model and applications, operations language, ...).

Key elements of middleware includes

- IPC (Inter Process Communication) handling communication between the workstations in the network including the broadcast mentioned earlier.
- HFA (History File Archiver) / cache, dealing with archiving of data and support of a cache to hold local data on workstations. The cache helps reduce access to the main archive and thus improves performance while lowering load.

The “Applications” cover all the classical areas of a control system. The influence of object-oriented techniques is seen in the split of a number of these into “model” and “applications”, where the applications in effect provide views, or processing, of the underlying model.

Some specific salient features of the actual architecture which has emerged are discussed below.

4.2 SPACECRAFT DATABASE

A relatively simple database approach was adapted based upon the following concepts:

- An on-line database comprising persistent (date) objects, supported by a special-to-project Object Store.
- Editors for the various categories of object (TM, TC, displays, etc.). These editors support consistency checking and locking, the latter permitting shared usage of the database. They work directly upon the database objects in the format they are used by the on-line system, although in practice a “draft” MIB is used for editing, followed by a “commitment” process (see next point).
- A version tool permitting selection of databases and switching a given “draft” database to operational status.

Here we should point out that the SCOS-II approach is quite different from most other mission control systems (including those at ESOC, MSSS and SCOS-IB). In the traditional systems, there is a “source” database and an “operational” database, in which the source database is offline and typically is maintained by a data management system (e.g. ORACLE for SCOS-IB). By contrast, in SCOS-II, the database (called the Mission Information Base, MIB) is on-line and consists of a set of persistent objects.

4.3 OPERATIONS LANGUAGE

The original requirements called for, in effect, a set of languages to be used in various areas of operational knowledge definition (data definitions, procedures). In practice only the scope of the language is limited parameters (of telemetry or commands) and in particular definition of derived parameters.

5. GROUND STATION INTERFACING: THE NCTRS

For interfacing to Ground Stations, a separate generic communications “front-end” is being developed. Rather than implementing this from scratch, the Network Control Transmission and Routing System NCTRS of the CLUSTER Mission is being reused and extended. In fact the NCTRS is being ported from its original host (DEC / VAXstation) to SUN / UNIX, with addition of interface software for new CCSDS packet TM / TC compatible ground station equipment as well as the older generation of equipment. This is proving a very cost-effective approach, since the CLUSTER project, with four spacecraft to support (using four control systems) had already developed a rather general facility, albeit predominantly on the older communication protocols (X.25 with a limited ESA private protocol on top). A particular feature of the generic NCTRS is that it supports the newer communications protocols which the station equipment requires (OSI, CMIP). The generic NCTRS is planned for use with either of ESA’s MCS infrastructures (SCOS-IB or SCOS-II).

6. SCOS-II CLIENT PROJECTS

SCOS-II is being used as the basis of the control system for a number of so-called “client” projects. These are HUYGENS, MARECS, Launch and Early Orbit Phase of the Meteosat Transition Programme (MTP). SCOS-II has also been used to complement the control system of SOHO, primarily in the area of historical TM retrieval and monitoring. Implementation of a SCOS-II based ENVISAT system was also started, but for reasons explained in section 8, this has been abandoned. Nevertheless, a number of interesting steps forward were made on ENVISAT, so this is reported in section 6.3.

The client projects have greatly helped in the development of SCOS-II:

- They have led to focussing of the development on their needs and have helped drive the critical selection of user requirements.
- They have exercised the SCOS-II deliveries, identified problems and provided pointers towards design weaknesses, which have subsequently been addressed by the SCOS-II project.
- In some cases, they have developed software which is potentially generic and which has then been fed back into the SCOS-II development.

6.1 HUYGENS

HUYGENS is the ESA provided element of Cassini / HUYGENS, the joint NASA / ESA mission to exploit the Saturnian System. Titan, the largest moon of Saturn, is the observation target of the mission. The spacecraft will be launched in October 1997, with HUYGENS attached to the orbiter. After an almost 7 year journey, the HUYGENS probe will be released from the mother spacecraft in late 2004. It will enter the atmosphere of Titan in order to parachute a fully-instrumented robotic laboratory down to its surface. During the cruise phase to Saturn, in-orbit check-out of the Probe subsystems will be undertaken every 6 months. The HUYGENS Monitoring and Control System (HMCS) interfaces the Cassini Ground Segment at JPL through a NASA provided Science Operations and Planning Computer (SOPC) installed at ESOC.

HMCS is responsible for

- planning, pre-validation and submission of command schedules
- retrieval and analysis of telemetry
- handling of external interface to NASA
- extensive support for on-board software maintenance

The HMCS is designed to run on a single UNIX workstation (“SCOS in a box”). It makes full use of the functional capabilities of SCOS-II, including middleware, monitoring applications (telemetry processing , displays, out-of-limits checking, retrievals, etc.) and commanding applications.

The on-board software maintenance function of SCOS-II has been developed by the HMCS team. It provides full capabilities to represent memory images, to interpret these according to database defined memory models, to compare memory images, process dump packets and generate patch commands.

Like SCOS-II, HMCS was developed using object-oriented analysis and design. The SRD and ADD were produced as delta documents to the corresponding generic SCOS-II documents. Training was given to the user community to understand and become acquainted with object-oriented terminology. The use of object-oriented analysis based upon a generic logical model has resulted in a higher level of re-use of SCOS-II elements than otherwise would have been the case. Mission-specific software has to a large extent been accommodated by changing existing SCOS-II software through use of inheritance.

Several problems arose during the development:

- Schedule. The HMCS schedule was very dependent on the SCOS-II schedule. This turned out to become a problem. However, risk management was continuously applied - hereby immediately identifying alternatives to overcome schedule problems. As an example, the HUYGENS configuration database system was developed by the HMCS team to relief the schedule dependencies.
- Documentation. The SCOS-II documentation was not up to date. Changes to SCOS-II interfaces were not communicated, creating the need for extra efforts on the HMCS development to accommodate these. A customisation guide was missing. A close dialogue was maintained with the SCOS-II developers to overcome this as well as to ensure that mission-specific changes would not cause problems with the SCOS-II system integrity.

The HMCS has demonstrated that SCOS-II can be successfully used as basis for a mission control system. It will be first used in validation tests with the probe at end 1996.

6.2 MARECS / MTP

6.2.1 MARECS

The mission control system for the telecommunications spacecraft MARECS is currently running on the Multi-Satellite Support System (MSSS), ESOC’s oldest MCS infrastructure. In late 1994 it was decided that this would be moved to the SCOS-II platform, in order to save maintenance costs (MSSS runs on rather old ENCORE/MPX hardware and in addition the MSSS application software requires a maintenance team). It was also considered that this would also provide a demonstration of the capability of SCOS-II to support an operational “in-house” mission.

The work involved was:

1. implementation of mission specific software: this included parts of the system which would not be expected to be provided by SCOS-II, such as TC uplinker and TM receiver;
2. implementation of special applications which are specific to MARECS (e.g. hard coded derived parameters, IRES blinding);
3. implementation of functionalities which had been expected from SCOS-II but were not in fact provided at the time of the conversion. e.g. centralised alarm and message handling, OOL display, TC history display, printouts, basic graphics. As a consequence workarounds had to be implemented for these missing functionalities. In fact, some of the functionality developed by the MARECS team (e.g. in the graphics display area) has been taken over by SCOS-II and further extended.

This last-named problem was in part the consequence of the MARECS conversion being carried out in parallel with intensive SCOS-II development activities. This also resulted in hitting a number of “teething” problems with new software: for example initial integration showed anomalous high loads for telemetry processing - this turned out to have several causes: inefficiencies in the first version of the operations language and inappropriate workstation configuration being two main ones.

A last point is that the MARECS team planned very little requirements analysis since the project was considered as conversion or porting exercise, since it was considered that the functionality was already known and documented. In reality this caused problems, since existing documentation was heavily biased towards the implementation on the legacy system (MSSS) and new team members has difficulty in differentiating such implementation specific artefacts from the problem domain functionality which had to be ported.

At the time of writing, the implementation is almost complete, with integration with the latest release of SCOS-II (Rel 1.2) ongoing. A demonstration to the users is planned for end July, which will be followed by several months of acceptance testing. The users have also been familiarising with an earlier version of the system during the last three months.

6.2.2 MTP

The MARECS system is also planned to be reused to support the LEOP of the Meteosat Transitional Programme. Data rates and functional requirements and TM/TC characteristics are very similar. The main difference is that a relatively large configuration of client workstations (~12) connected to one or two servers is involved.

6.3 ENVISAT

For ENVISAT, an essentially complete database subsystem was developed, as described in sections 6.3.1 through 6.3.3.

6.3.1 EXTENSION OF THE MIB

ENVISAT required extensions to many of the SCOS-II baseclasses. Inheritance was found to work quite well and the functionality provided by SCOS-II was easily reused even though some of the underlying classes were different.

6.3.2 DATA IMPORT FROM THE SPACECRAFT MANUFACTURER

The satellite reference database comes from the spacecraft manufacturer in a relational database format. This had to be converted into the OO format used by SCOS-II. This is performed by a complex piece of SCOS-II “import” software. In this area the SCOS-II code was taken and modified for the ENVISAT MIB.

6.3.3 EDITING THE MIB

Because the database system (Object Store) is proprietary, the editors are written in C++ using a proprietary user interface builder tool OI™. There is one base class editor which provides the virtual description of an editor and much common functionality. For each editable class (or in some cases, set of classes) an editor sub-class must be created along with an OI configuration file giving the layout of the editor. ENVISAT has also implemented some sub-classes of SCOS-II editors. Inheritance is heavily used resulting in good reuse of SCOS-II code. The editors are Motif-based and provide navigation among related objects using a “click and go” mechanism. ENVISAT made many improvements here which were fed back to benefit SCOS-II, the two major ones being multi-user editing and user friendly interface to MIB consistency checking.

Presenting the users with an editable version of the SCOS-II logical data model provided very useful feedback on the model itself. When the users could actually see the data and its relationships they were able to spot immediately problems with the intended implementation of TM and TC handling. This was far easier than for instance trying to spot the same problems as documented in an Software Requirements Document.

7. LESSONS LEARNED

Baldi et al (1995) have described the main lessons learned in the SCOS-II development. These cover software methods and standards, approach to implementing generic systems, project management, prototyping and distributed system design. The conclusions of this earlier paper remain valid and we will not go over this ground again. Instead we will look in some detail at specific aspects mentioned in this paper.

7.1 USER REQUIREMENTS VERSUS IMPLEMENTATION

One thing immediately apparent to the reader from Chapters 3 - 4 is the descoping from the original plans. A number of points should be noted concerning the user requirements:

1. The more advanced features mentioned in Chapter 3 had never been subjected to any feasibility trials (e.g. in "Phase A" type studies). Thus feasibility of implementation and demonstration of actual operational potential had not been carried out. The latter point is important: does the user community really want each new feature specified in the URD?
2. Work on the user requirements went on in parallel with the development of the system. Thus for example Issue 4 of the URD was released in late 1995, after Release 1 of SCOS-II had been made. This led to unclarity about baselines and pressures for late changes
3. The URD was itself a long, complex document written in a rather abstract way. It has over 5,500 requirements, with no prioritisation or layering of requirements. As a consequence it was difficult for operations staff other than the authors or software engineers to understand it.

The lessons to be learnt from this are

- the URD baseline needs to be fixed quite early in the project.
- any new operational features or concepts must be prototyped early (before the URD is "frozen") and validated by the users.
- the number of requirements should be limited and "layered", so that parts to be excluded from the scope of the development can be easily identified.

7.2 PLANNING OF INFRASTRUCTURE DEVELOPMENTS AND CLIENT MISSIONS

It is apparent from some of the problems mentioned under client missions (e.g. MARECS, HUYGENS) that quite a lot of difficulties resulted from the parallel development of infrastructure and the client missions on which it is based.

Here there is no easy answer, since it is clear that an infrastructure has to have "customers" to make it pay off and also to ensure that development priorities are concentrated on real mission needs (SCOS-II client missions have had this beneficial effect on SCOS-II). On the other hand, this means that some missions have to take a certain risk in using a new infrastructure for the first time - if no one will take this risk, there will never be any change!

The answer is probably to a) ensure that there is adequate exploration of the new technologies in a suitable product oriented preparatory programme b) exercise careful control of requirements, as mentioned in section 7.1

and c) make the decision for major projects only when an adequate kernel of the infrastructure is available and validated.

8. STATUS

Very recently, a design consolidation exercise of SCOS-II was completed. In parallel to this a Firm Fixed Priced offer was requested from Industry, for completion of SCOS-II. This was issued based upon a carefully selected subset of the SCOS-II URD. In the event, it has turned out to be impossible to continue the project on this basis, since severe cuts in ESOC's budget 1996 - 97 meant that the necessary completion funding was no longer available.

Given the large amount of implemented software, the following approach was decided upon by ESOC Management:

1. The viability of SCOS-II in its present release will be validated with three projects: HUYGENS, MARECS and MTP, for which development is largely completed.
2. Two other major projects ENVISAT and XMM will be based on the older infrastructure (SCOS-I): these would have needed the full SCOS-II, which owing to the budget cuts could not be made available in the necessary time frame.
3. Completion of SCOS-II according to the available consolidated design is planned to be done under a Technology Development programme. It is foreseen that this will be split into two parts: (1) SCOS-II completion and (2) pilot project. The pilot project will be a parallel implementation on a complex project already under development on an older infrastructure.

REFERENCES

Baldi, A., Head, N., Jones, M., and Kaufeler, J-F, 1995, *Lessons Learned from Development of SCOS-II, a Reusable Spacecraft Control Infrastructure*, Symposium on Reducing the Cost of Spacecraft Ground Systems and Operations

Baldi, A. & Pace, M., 1994, *A New Communication Protocol Family for a Distributed Spacecraft Control System*, *Proceedings of SpaceOps 94*, Greenbelt, MD.

Jones, M., Head, N., Keyte, K., Howard, P. and Lynenskjold, S. 1994 *SCOS-II: ESA's New Generation of Mission Control System*, *Proceedings of SpaceOps 94*, Greenbelt, MD.

Kaufeler, P., Pecchioli, M. and Shurmer, I., *SCOS-II: ESA's New Generation of Mission Control System, The User's Perspective*, *Proceedings of SpaceOps 94*, Greenbelt, MD.