SAFETY ASPECTS OF SPACECRAFT COMMANDING

Mr. N. Peccia
European Space Operations Centre (ESOC),
Robert-Bosch-Str. 5, 64293 Darmstadt, Germany

ABSTRACT

The commanding of spacecraft is a potentially hazardous activity for the safety of the spacecraft. Present day control systems contain safety features in their commanding subsystem and in addition, strict procedures are also followed by operations staff.

However, problems have occurred on a number of missions as a result of erroneous commanding leading in some cases to spacecraft contingencies and even to near loss of the spacecraft. The problems of checking commands in advance are increased by the tendency in modern spacecraft to use blocked/time-tagged commands and the increased usage of on-board computers, for which commands changing on-board software tables can radically change spacecraft or subsystem behaviour.

This paper reports on an on-going study. The study aims to improve the approach to safety of spacecraft commanding. It will show how ensuring "safe" commanding can be carried out more efficiently, and with greater reliability, with the help of knowledge based systems and/or fast simulators.

The whole concept will be developed based on the Object-Oriented approach.

Keywords: **Telecommanding, Safety, Predictive Knowledge, Object Oriented**

1. INTRODUCTION

This paper gives an interim report on a study of the safety aspects of spacecraft commanding. The overall aim of the study is to demonstrate the feasibility of model-based command checking.

The study examines user requirements for such a system. Based upon these requirements the functional requirements and the architectural design is being produced. Finally a prototype of at least the basic mechanisms of the design will be developed and demonstrated.

The whole concept will be developed based on the Object-Oriented approach. The common environment must provide the different spacecraft users with the same kind of user interface facilities in order to offer a consistent operational environment.

The ESA SCOS II system (under development) is being taken as the reference system to be interfaced. SCOS II will operate in a hardware and basic software environment that is vendor-independent.

The function of a SCOS II (Spacecraft Control Operations System) system are seen as a collection of independent models of various parts of the spacecraft and the ground segment. SCOS II will therefore provide a library of 'building blocks', which can be combined in various ways to produce the overall model. To allow this to be done easily,

object-oriented software engineering technology has been updated for analysis and implementation of SCOS II. Specifically the Coad/Yourdon method and the C++ programming language have been chosen.

Not all missions are the same, which led to make modifications to the library building blocks to be used in a specific mission. Using an object-oriented technique known as 'inheritance', it will be possible to provide a customised building block for a given mission, whilst maintaining the same interface.

The SCOS II system will be hosted on a Local Area Network (LAN) of distributed UNIX workstations. Some centralised services of the system will be provided by server processors (client-server concept). The use of a distributed system also offers advantages in terms of system availability and failure tolerance.

An initial delivery of the SCOS II system is foreseen for end 1994. It will contain basic functions of the system. The Huyghens-Cassini, Envisat and XMM spacecrafts will make use of the SCOS II infrastructure software.

2. BACKGROUND

2.1 CURRENT STATUS

It is useful to describe first the general ESOC approach to handling of commands by the Mission Control System (MCS) for currently supported missions, which however can be significantly modified for specific missions.

- Command Preparation Checking

  In the command database to determine allowable ranges of parameters, etc.

  Automatic checks on "manual or automatic stacks of commands" at time of entry of command parameters.

- Pre-Transmission Validation (PTV) of commands

  The normal route for all commands involves a pretransmission validation (PTV) before the command is passed to the ground station for uplink. PTVs are defined in the command database.

  Checks normally performed in PTV are:

  TC configuration (e.g. check that the TC subsystem has not been disabled)

  Spacecraft and subsystem status, as computed from incoming telemetry parameters. The TM parameters and the mode computation are specified in the command database. PTV can be disabled by the operator and by the command source. PTV does not provide for limit checking or other checks of individual command parameters or of parameters sets.

- Checking of command contents

  This is not a standard facility on the ESOC Mission control system; it varies from one mission to the other. Any such checks performed are limited since :

  They are only static limit checks (e.g. lower and upper limits) on individual parameters.

  Many commands cannot be checked against fixed limit checks alone because of interdependence between

parameters.

The correctness of multiple command activities cannot correctly be checked.

Command parameters are obviously important parts of a command and for some commands the value of the parameters can be vital for the spacecraft safety.

No on-line checking of combination of commands and command parameters nor pre-execution validation of commands against predicted spacecraft status is carried out or envisaged for current "in flight" or near future missions (ERS-2, ISO, CLUSTER)

2.2 PLANNED DEVELOPMENTS

Future missions to be supported by the ESA SCOS II (under development) will be controlled using approaches to commanding which are likely to differ significantly from the current one. Special services should be provided to increase the safety of commanding. Two additional types of conditions will be used in making these safety checks :

- a predicted set of conditions in the on-board status applicable at the (future) time of execution (and not necessarily at the time of release)
- a set of "operational constraint" rules to be obeyed following command execution.

These checks are carried out based on a prediction of the on-board status at the planned execution time (Predictive Knowledge). Thus a capability to propagate the on-board status needs to be available for all the potential sources of commands (Manual Command, on-board Master Schedule and ground automatic command files).

Predictive Knowledge allows the prediction of future states of the system under control (satellite modes, measurements, etc) from a "known initial state" and taking into account planned commanding activities and predicted mission events.

This Predictive Knowledge can be produced in two ways :

- Evolution of the system in the absence of any commanding activity (Evolution Predictive Knowledge)
- Evolution of the system under the influence of Telecommanding (commanding Predictive Knowledge).

In addition , detected or predicted on-board autonomous actions can be treated in an analogous manner to telecommand actions. Specific attention shall be given to the handling of asynchronous on-board actions (these are often the result of failures and related on-board corrective actions).

This knowledge may be in the form of algorithmic, heuristic or mathematical models. The predictions will be required both over a short term (e.g. for satellite health monitoring) and over a long term (e.g. to validate a plan spanning several days).

3. OVERALL APPROACH

The study has the following steps :

- Problem, methodology analysis and evaluation of the ESOC requirements

- Software Requirements Phase

- Architectural design of the system

- Prototyping and demonstration of the basic design

4. BASIC REQUIREMENTS

The central idea is the use of a Model of the satellite. The definition of this Model of the spacecraft is the most critical part of the study. It is of course of major importance that the real system is modelled as close as possible. The Model has to run quickly to allow predictions for some time in the future (typically 48 hours for EURECA) in case of on-board time-tagged commands checking.

During operations this Model must be capable to be connected to (or be a part of) the spacecraft control system whereas during the validation phase to the Expert Tool system for FOP (Flight Operation Procedures) production. The following scenarios are considered :

a. "On-line" : The Model is part of the mission Control System (SCOS II), and each command is checked (.e.g for consistency with the modelled "image" of the spacecraft) before being released for uplinking to the spacecraft.

b. "Near Realtime" : The set of commands to be sent to the spacecraft (either from Manual Command or Automatic Schedule) are previously uplinked (or could be done "directly" by the system) to the Model respecting the "timelining" (timing and ordering of activities). This should allow the user to view the changing state of the Model while it is being "operated" and will also perform concurrent safety checking and validation of the operations in each scenario exercised.

The command validation function (in the Model) should use the Predictive Knowledge of the impact of the command (together with any other planned or predictable actions) to cause the rejection of a TC based on predicted effects which violate any health criteria. This information will be passed to SCOS II, which will inhibit the uplink of the command.

During Planning validation (sequence of commands as output of the mission planning) it will normally be necessary to propagate the mission state during the planning interval in order to :

- establish that pre- and post-conditions for activities are fulfilled

- to confirm that health criteria are continuously satisfied during the planning interval

c. "Off-line": User selected Flight Control Procedures (FCP), Contingency Recovery Procedures (CRP) or timelines shall be applied to the Model in order to validate the operations (Procedure Validation).

The following Model operating scenario could be envisaged :

- The Model is initialised with the available TM in order to synchronize the its internal state with the real state of the spacecraft.

- As a second step the Model is let to evolve by means of a prediction generation function, taking into account the planned on-board mission events and / or commanding activities.

The Model could also be used as follows:

- Verification of commands executed in the past (e.g. comparison of playback telemetry and predicted mission status)

- Monitoring functions including the display of predicted telemetry parameters during "non visibility" periods.

- Diagnosis : The deviations of predicted values from the expected ones could be detected and analysed. To this aim a knowledge not completely contained in the Model is required (e.g. diagnosis charts and fault trees contained in the spacecraft Operations Requirement Handbook)

The Model is a central concept on this study. It predicts mission states related to future mission times. The selected approach is based on two types of model :

- A complete Model for near real time and off-line scenarios

  Detailed spacecraft subsystems models are developed at ESOC for each mission, as part of spacecraft dynamic simulators used for validation of control system software and Flight Control Procedures as well as for staff training. This type of simulators run 30 times faster than real time when running on an ALPHA VAX platform.

  The Model is extracted from an existing spacecraft simulator. It shows the best precision in the states prediction in spite of a lower speed. For this reason it will be used when greater accuracy is required.

- A simplified Model using knowledge based techniches for real time scenarios

  High speed performances are met but a lower accuracy in the computation of predicted states is shown. The Model is build up extracting the mission information from a selected repository (e.g. the Mission base in SCOS II) and adding manually the missing information.

This two Model approach should be used for model validation. In order to trust such a system strong emphasis should be put into the verification and validation of the models.

5. SOFTWARE REQUIREMENTS

The Software Requirements Document defines the functional requirements of the system according to the SCOS II Development Standards.

The document covers the system functionality, outlines standards for input and output data which they should handle, and shows how they should interface to the wider operational environment in the future.

The whole concept is being developed based on the Object Oriented approach. The expected benefits of OO for the Model of the spacecraft are :

- natural modelling of the architecture of the spacecraft

- flexibility (via properties of inheritance and polymorphism)
- different levels of abstraction, permitting viewing of the Model at different levels of complexity
- potential of reusability

The design and implementation of the system should support the Object Oriented Paradigm. The system should interface with SCOS II and should be based on "open architecture" so as to allow for additionally functionalities via added modules.

The system has to be based on UNIX, and developed and maintained on SUN platforms. However it will be capable to run on any of the main line of available UNIX platforms (e.g. SUN, HP, IBM and Digital).

The main constraints are the following:

- The system should access the SCOS II Mission Information Base to derive the Predictive Knowledge, the operational constraints and the execution verification criteria. The user should not insert significant additional information.
- The system should not cause detectable performance degradation on SCOS II real operations.
- The system should have the capability of synchronizing its internal Model status with the real spacecraft data and status.

After an Object Oriented Analysis of the system the following OO diagrams were produced :

- Model OO Diagram

  It focuses both on the Model related abstraction level and on the high level internal decomposition of the system. The two Model approach is introduced as a keypoint in the whole system organization. A "complete" Model cooperates with a "simplified" one to obtain the best performances in terms of accuracy and computation speed.

- Database level OO Diagram

  It shows the database internal organization focusing on the elements needed to build the Model (e.g. system element, activity, application criteria of system elements, verification and validation criteria of activities)

- Operational Context Diagram

  It describes the different operational scenarios, particularly the real time case which is the most complex one

The following interfaces are envisaged:

- SCOS II command stacks (e.g. manual and automatic stacks)
- SCOS II Mission Implementation Base
- Display of system outputs on SCOS II Man Machine Interface
- Telemetry acquisition from SCOS II telemetry Processor
- Flight Operations Procedures Set Tool to read and process Flight Operation Procedures in the off-line case
- Model of an existing spacecraft

simulator to be used as the "complete" Model

6. CONCLUSIONS

At the time of writing this paper (July 1994) the Architectural Design Phase is in progress. This phase defines the architectural concept, considering all functions and also how the system should support future expansion and modification of functionality. The Architectural Design Document should include detailed descriptions of all critical design elements, such as data storage architecture and access methods, control data structures, knowledge representation and all external data interfaces.

During a second phase the study should produce the following :

- A detailed Design and implementation of a prototype. A spacecraft subsystem should be identified to develop such a prototype (a partial Model). It will be integrated with the SCOS II system at ESOC
- A Detailed Design Document (DDD) of the prototype
- A Software User Manual (SUM) of the prototype

This study aims to produce a prototype to improve the approach to safety of spacecraft commanding by using model-based command checking systems. This philosophy can then be used for upcoming ESA missions such as those of XMM and Integral.

7. REFERENCE DOCUMENTS

1. ESA Software Engineering Standards, ESA-PSS-05-0 Issue 2, February 1991
2. SCOS II Development Standards, SCOS II-CON-001, Issue 1, January 1993
3. SCOS II User Requirement Document, Issue 1, March 1993
4. SCOS II Software Requirement Document, Draft 2, May 1993
5. SIMSAT Simulator Designers Manual, Issue 1, May 1993
6. Study "Safety Aspects of Spacecraft Commanding" Problem Analysis and Concept Definition, Dataspazio, March 1994
7. Study "Safety Aspects of Spacecraft Commanding" Software Requirements Document, Dataspazio, June 1994
8. "Object oriented analysis", Peter Coad & Ed Yourdon, 2nd edition, Prentice Hall, 1991