Use of an Operations Simulator for Small Satellites

Sid Saraf, Chahé Adourian, Philip Melanson, Mak Tafazoli

Canadian Space Agency
Space Technologies Branch, Ground and Flight Computers & Simulations Section
6767 route de l'Aéroport, St-Hubert, Québec, Canada, J3Y 8Y9
Tel: (450) 926-6576, Fax: (450) 926-4695
sid.saraf@space.gc.ca
www: http://www.space.gc.ca

Abstract. The Canadian Space Agency (CSA) has initiated the Science Satellite (SCISAT) mission as part of its ongoing space science program. The SCISAT-1 satellite will be operated from CSA's mission operation centre in St-Hubert, Québec. The use of an operations simulator is critical in mitigating any mission level risk. During an anomaly situation the operation team's only line of defence against a mission failure could be the simulator. The SCISAT-1 simulator could also be an effective tool to ensure that commands or command sequences that are detrimental to the spacecraft or the science planning are not up-linked accidentally. The best argument for the need of a simulator is encountering unknown scenarios that cannot be tested before launch.

Due to the budget constraints of a small program, the fidelity of the simulator may have to be compromised to ensure critical capabilities that maximize risk mitigation while keeping the cost of development and maintenance low. This paper will describe the uses of the simulator for such a mission and the criteria that were used in selecting the simulator hardware and software in order to meet the requirements. The correct development choices allow the reuse of simulator software for future micro-satellite and small satellite programs. Therefore, the knowledge and resources gained will distribute the simulator cost over many years. In addition, the lessons learned from this project will allow CSA to absorb programmatic risks initially before the knowledge and expertise can be passed on to industry for future missions and managed effectively by CSA.

Introduction

The SCISAT-1 mission will include new technology that is being developed in Canada and will be applicable to other Canadian micro and small satellite missions. Bristol Aerospace is the prime contractor for the bus. The Atmospheric Chemistry Experiment (ACE) from the University of Waterloo has been selected for the first SCISAT mission and will be launched in the third quarter of 2002 on a Pegasus XL launch vehicle. The main goal is to measure and understand the chemical and dynamic processes that control the distribution of ozone in the upper troposphere and stratosphere. The satellite is designed to operate in a 650 km, 74 degrees inclination orbit, for a period of two to five years. A simulator is intended to support the operations of SCISAT-1, which will be based at the CSA in St-Hubert, Québec. The limited experience in developing, launching, and operating a small satellite mission makes the requirement of an operations simulator more important compared to operators that have flown many small satellite missions. This paper will cover the intended use of the simulator, outline the simulation models, describe the hardware/software selection process, and discuss the implementation of the simulator into a multi-mission operations center.

SCISAT-1 Atmospheric Chemistry Experiment Simulator

The main purpose of the Scisat-1 ACE Simulator (SAS) is to provide the CSA Space Operations team with a reliable means of procedure development & mission plan validation, personnel training, anomaly isolation & resolution, bus software maintenance & testing, off-line sub-system simulations, ground segment verifications, and potentially payload scheduling assistance and verification. SAS will have to be versatile enough to support the mission during pre-launch preparations, launch and early operations, commissioning, routine science, new technology demonstration, and end-of-life operations. A diverse range of personnel including new hires, spacecraft engineers, specialized subsystem analysts, simulation campaign directors, procedure developers, and simulation conductors will use the simulator.

Table 1 describes how and when SAS will be used in support of the SCISAT-1 mission [1].The mission is broken down into its phases (columns in the table) including mission preparations, Launch and Early Operations (LEOP), On-orbit Commissioning Phase (OOCP), and Science Operations Phase (SOP). Each

row in the table represents a major category of simulator use. The cells in the table describe how the major categories of simulator use will be implemented during each mission phase [2]. Table 2 describes the categories from Table 1 in further detail. Figure 1 below provides and overview of interfaces between the users and SAS through the Ground Control Data Systems (GCDS) [3]. The SAS functional areas are shown by the circles, the icons show external interfaces, and arrows show the data control and flow.

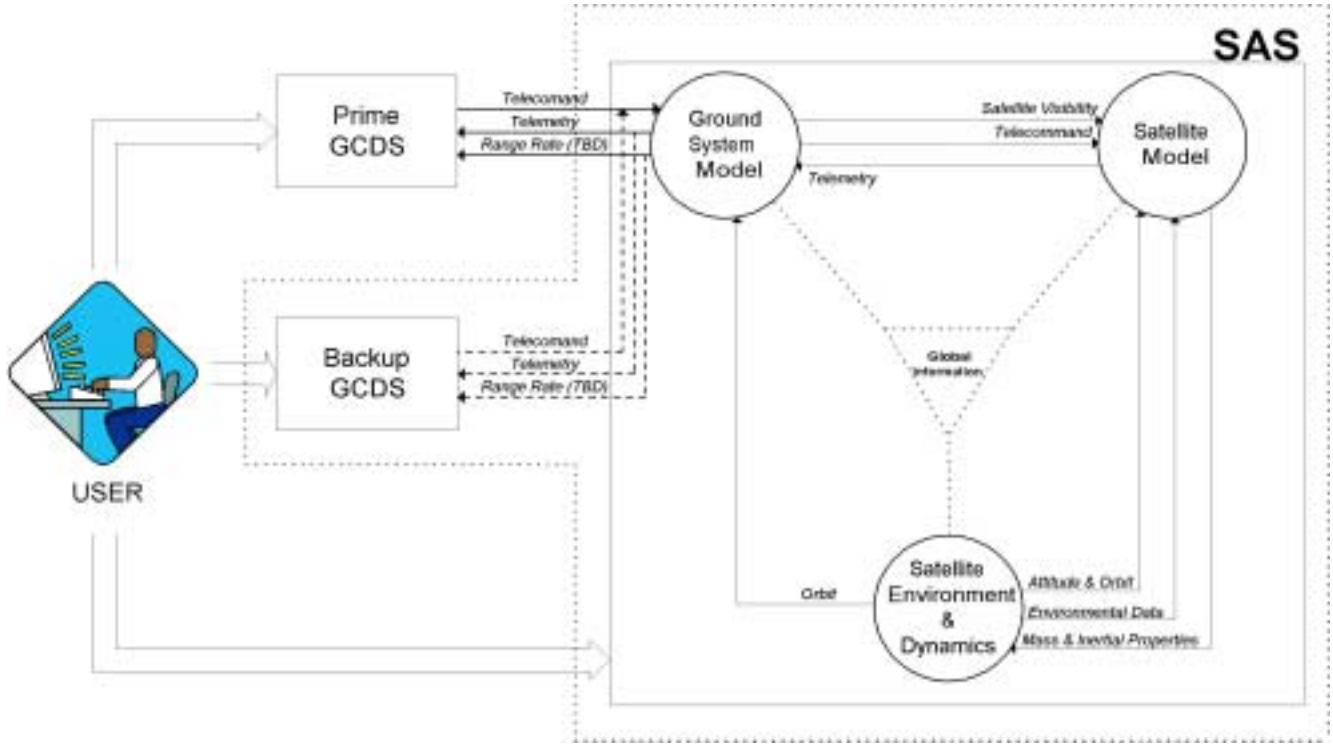

Fig 1. SAS Functional Diagram

Table 1: SAS Use During SCISAT-1 Mission Phases

	Preparation	LEOP	OOCP	SOP
Procedure Validation	Baseline procedure validation	New procedures resulting from anomaly	Same as the previous phase	Same as the previous phase
Rehearsals & Training	Simulation campaign (training of all staff)	In the event of anomaly and only some staff	Same as the previous phase	For contingency, new staff or same as for the previous phase
Anomaly Isolation & Resolution	Not applicable in this phase	If an anomaly occurs	Same as the previous phase	Same as the previous phase
Bus Software Maintenance & Testing	Not applicable in this phase	If any anomaly requires bus software change	Same as the previous phase	Same as the previous phase
Off-line Subsystem Simulations	Not applicable in this phase	If any anomaly occurs	Spacecraft reconfiguration	Same as the previous phase
MOC Verification	MOC verification and readiness	Not applicable in this phase	Not applicable in this phase	In case of GCDS reconfiguration

Table 2: Details of Simulator Use

	Procedure Validation	**Rehearsals & Training**	**Anomaly Isolation & Resolution**	**Bus Software Maintenance & Testing**	**Off-line Subsystem Simulations**	**MOC Verification**
User	Operator/ Procedure developer	Mission Operations Centre**	Operator/ Analyst	Operator/ Engineers and Analysts	Analyst	Operators
Duration	Hours per procedure (depends on procedure complexity)	No. of passes for event to occur (e.g. several passes or a single pass)	Variable	Days	Hours	Days
Data Timing	Real-time or Off-line	Real-time	Off-line	Off-line	Off-line	Real-time
Simulation Conductor Actions	If required for non-nominal steps, works interactively with the developer to configure the simulator to execute procedure. Likely to take several iterations	Works independently to configure the simulator for the event; injects anomalies as needed; monitors the real-time interactions among the staff	Works interactively with the analyst to configure the simulator; likely to take several iterations	Works interactively with the engineers and analysts to configure the simulator to test bus software	Works interactively with the analyst to configure the simulator	Works interactively with the operators to configure the simulator to verify GCDS, likely to take several iterations

**The scope of the group participating in this type of simulation varies according to the type of training required.

Simulator Development Environment

Many innovations arise from the low budget and short development duration requirements set forth for Scisat-1 ACE and similar small satellite missions. Any simulation tool being considered for such a mission will have to facilitate a fast turn-around development and validation time while adhering to stringent fidelity and reliability needs. Such a simulator will have to include model source code accessibility, design flexibility, user friendliness, and real-time as well as faster than real-time capabilities. It will also have to have an open architecture and facilitate complete document generation. In order to maximize return on investment, the simulator technology will have to be easily cloned for other spacecraft missions. The simulator development tools were selected based on a study comparing various products using an extensive criteria list. The following 6 tools were evaluated: CAE ROSE (www.cae.ca), ESA's SIMSAT-NT (www.esa.int), FokkerSpace's EuroSim (www.eurosim.nl), Mathworks' Simulink (www.mathworks.com/products/prodoverview.shtml), Wind River's MATRIXx (www.mathworks.com/products/matrixx) and Boeing's Easy5 (www.boeing.com/assocproducts/easy5). Out of these, CAE ROSE was eliminated since the shrink-wrapped version will not be supported in the future and would not evolve. MATRIXx was eliminated since support will be discontinued in two years. Simulink was selected over Easy5, for its wide use and the availability of many popular toolboxes (Matlab code) and blocksets (Simulink modelling blocks) for a wide range of applications, including robotics, communications, ACS (Attitude Control System), orbital dynamics, etc. The list of available add-ons is extensive. Boeing's Easy5 was also thought to be a very good contender, and it may even have some capabilities not available in Simulink, however it lacks widespread use and the availability of add-on toolboxes is limited. The available libraries (similar to Simulink's modelling blocks) are very powerful and if the project involved their use, Easy5 would have been a good choice. Although Mathworks products provide a very good environment for both prototyping work and generation of real-time code, it was not thought to be sufficient to develop satellite simulators. For that purpose, there were two remaining possibilities: EuroSim or SIMSAT-NT. The idea being that many of the required models could be developed using Mathworks products (Simulink), generate C-code using the Matlab compiler and the Real-Time-Workshop, then use it in either EuroSim or SIMSAT-NT.

The choice between EuroSim and SimSAT-NT was not easy, but in the end, EuroSim was thought to be a more mature and capable environment. Another reason for EuroSim was the existence of MOSAIC: a tool that converts Simulink and Stateflow models into EuroSim models. EuroSim comes in two forms: a version for the IRIX/SGI platform and another for the Linux/Intel platform. The SGI version has been available for several years, it has more features, and it is the version used by major European Aerospace companies and agencies. The more recent Linux version seems to be working well, although it lacks some interesting features found in its IRIX counterpart (such as an integrated HLA support for distributed simulations). One of the beneficial features of EuroSim is the ease of incorporating C-code into the simulation. In most cases, there will be no need to modify the code before integration with the remainder of the models. In SIMSAT-NT, the code has to be implemented as a DCOM model (Microsoft' DCOM technology). This is not a straightforward process, nor is it easy to learn. Also, this necessitates the use of NT, which is not a very appropriate operating system for real-time applications. From a scheduling perspective, EuroSim has hard-real time capabilities (not available on Linux, but planned for future versions), multiprocessor support, and a very intuitive and powerful graphical interface to design the schedule.

Some of the major EuroSim features include:

- Operations simulator development tool with good documentation and support;
- Wide spread use by many European organisations for space related activities;
- Open system architecture which enhances portability of model software;
- Graphical interface to integrate model code into simulator;
- Graphical interface for advanced simulation schedule specification;
- Run-time scheduling of events and execution of intelligent scripts;
- Task execution time profiler;
- Simulation model code run real-time without modification;
- Facilities for model development, simulator composition, simulation preparation,

simulation execution, results analysis and configuration control;
- Built-in facilities to test the models used in the simulation;
- Scalability through easy code parallellisation and network distribution;
- Incremental replacement of software models by their corresponding hardware elements is facilitated;
- Client-server architecture allows to start/control/monitor multiple simulations on multiple platforms from a single interface;
- Hard real-time capabilities;
- Hardware and man in the loop support.

The use of the Simulink graphical environment to develop the subsystems and of MOSAIC to port the model into EuroSim has many benefits:

- Matlab, Simulink, and Stateflow are widely known and used products: more than half a million users worldwide;
- Graphical representation of models facilitates the documentation, testing, debugging, modification and peer-review processes;
- Automated tools can generate documentation from Simulink/Stateflow models;
- Environment ideal for fast prototyping work;
- Availability of hundreds of add-on products: Matlab Toolboxes, Simulink blocksets, and third party tools. These include many space-related products, including toolboxes and blocksets for orbital dynamics, attitude control, communications, and creation of fast thermal models;
- Existence of these toolboxes and blocksets enhances and speeds the development process by relying on extensively used and tested code. It also encourages reuse;
- Enables a top-down design approach: the high-level architecture is specified directly in Simulink through the use of system blocks;
- Interface between different system blocks is specified in the high-level architecture, which is used by all developers;
- Integrating the different system blocks together is less prone to errors since these blocks had to conform to the predefined interfaces in the high-level design;
- The Model developer does not have to be an expert in programming, only an expert in his field and have a good knowledge of Simulink;
- Analysts/System experts benefit from this approach since the models they are interested in were developed in Simulink. This allows them to carry out their analysis in one of the best environments available for that purpose and using only those models they are interested in;
- When analysts/system experts modify models and require these to be ported into the simulator, the normal simulator development process is followed: MOSAIC converts these models into EuroSim models and these models are loaded into EuroSim. This minimises errors introduced during the porting process;
- TLC (Target Language Compiler) files specify and control the code RTW (real-time workshop) generates from Simulink models. As such, the generated code for EuroSim is fully customizable;

Spacecraft Models in SAS

The level of fidelity of the simulator can be described in detail by looking at the specifications of each subsystem module. In order to reduce costs, some models have been ported from other projects and have been modified for the SCISAT-1 mission. In addition failure requirements have been kept at a minimum since their validation and testing can increase cost significantly. EuroSim, the selected simulation tool (selection process is described later), permits organization of code in different modules. This way, each subsystem can be individually coded rather than interlaced together. This makes each subsystem flexible, interchangeable, and reusable. Following the approach of reducing design costs, negotiations are under way to use the bus flight software source code to develop the Command and Data Handling (C&DH) subsystem module. Fast integration will be achieved by taking this source code and by removing the real-time hardware dependencies. Each subsystem could be modeled in a simple or detailed approach. By carefully keeping in mind the budgetary objectives of the mission as well as the fidelity of SAS, the requirements for each subsystem model where specified. The following paragraphs provide a brief description of each subsystem[2].

Command and Data Handling subsystem(C&DH)

The objective of the C&DH is to interface with all other subsystems on the spacecraft. Therefore, the C&DH must forward telecommands to all other subsystems and gather telemetry from them.

Other C&DH subsystem functionality: Control (activation / deactivation) of all subsystems; Satellite timekeeping as well as health and safety management; Fault recovery.

C&DH subsystem failures: Software failures such as timeouts, single event upsets (e.g. bit flips), software upload errors; Hardware failures such as parity errors, memory protection errors, memory bank failures; Other Failures such as subsystem interface failures, on-board clock failure

Attitude Determination & Control Subsystem (ADCS)

The ADCS subsystem is of high importance. Orientation of the spacecraft is the basis for the science operation mode. Apart from the control algorithm (which will be included in the flight software), all ADCS components will be modelled such as torque rods, momentum wheel, gyro wheel, sun sensors, etc. Failures will be possible on all components.

Power Subsystem

The power subsystem's main objective is to model the current and voltage applied to other subsystems. The solar panels, the power generated by them, and the batteries will be modeled.
Other power subsystem functionality: Simulation of battery charge level, charge/discharge characteristics; Power generation reactions to solar flux with seasonal variation, switching function

Power subsystem failures: Spurious switching; Bus over-current; Bus under-voltage; Battery overcharging and cell failures; Low battery charge

Thermal Control Subsystem

This subsystem will consist of a simple model that will simulate the thermal control subsystem's response to commands such as heater switching and attitude changes. Environment variations such as solar flux will also be accounted. In addition SAS shall simulate the effects of all of the thermal hardware such as heaters, thermal radiators, thermal coating, etc.

Thermal Subsystem failures: Heater failure; Radiator failure; Thermistor failure; Degradation of thermal properties over time.

Communications (Comms) subsystem

The Comms subsystem is responsible of receiving and transmitting real-time data at required rates. The model must simulate the command database, the receiving S-band uplink and demodulation, CCSDS-compatible digital bit-stream downlink and uplink as well as spacecraft antenna switching, filtering, and combining.

Comms subsystem failures: S-band RF antenna; Command error (parity, receipt check, incorrect command formats, spurious commands); Receiver carrier levels; No demodulation lock

Science Payloads (FTS & MAESTRO)

Two scientific payloads will be on board SCISAT-1: FTS and Maestro. Each module will be modelled so that the behaviour of each can be visible in telemetry for all modes of operation. Therefore, the housekeeping telemetry coming from both modules shall be sufficient to determine their behaviour. Science data will be provided from static dummy sample files.

SAS Integration in the Satellite Operations Centre

In operating multiple satellite missions, the CSA has begun studies to create the Canadian Satellite Operation Centre (CANSOC) [4], a multi-satellite operations facility [5]. CSA is currently operating the shelf (COTS) products in order to reduce mission costs. In this perspective, the SAS must be incorporated not only to achieve an appropriate level of fidelity but also to integrate it successfully with the rest of CANSOC. SAS must be a finite element able to come to life rapidly, to provide simulations through prime facilities and also provide simulations through secondary equipment. This will enable it to operate during other activities in CANSOC. Fig. 2 describes the key elements of SAS and how it will be integrated in CANSOC.

The Prime TT&C station, which consists of an antenna, RF converters and digital processing unit is linked to the telemetry processor (PTP-NT) which acts as a front end to the computer network as well as de-commutation of telemetry and commutation of telecommands. Finally the control stations to be used by the controllers are separate workstations running the Integrated Test and Operations System (ITOS). These components will connect to the PTP-NT via TCP/IP connections.

SAS can be incorporated CANSOC with minimal interference due to its position in the data flow. During real-time passes of a satellite, SAS will have no interaction with the PTP-NT. If any simulations were needed during this period, they would be conducted from backup GCDS workstations. During the periods when no real-time operations are being conducted, SAS can provide simulations by connecting to the PTP-NT. In this scheme the satellite operators can use the same workstations and interact amongst themselves as they would during real-time operations. Having

RADARSAT-1 and plans are in progress for the operation of RADARSAT-2 and SCISAT-1. Thus, these new missions create the need to develop CANSOC. CANSOC will make use of existing resources and will maximize the use of commercial off

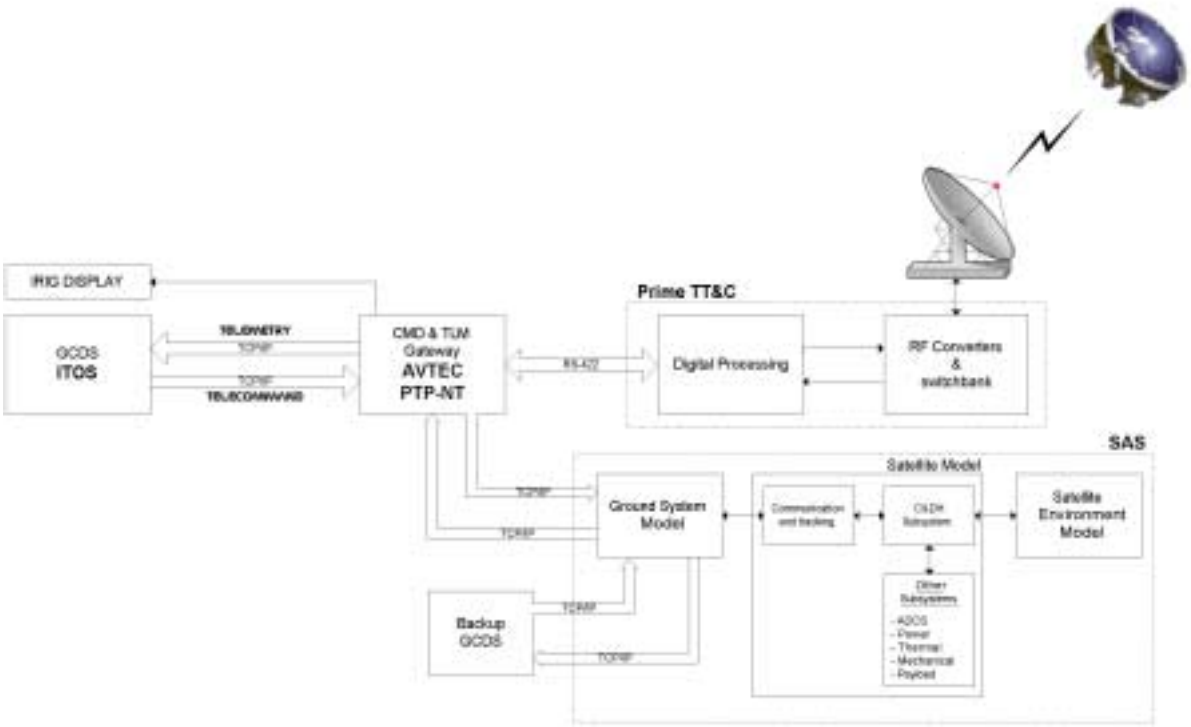

Fig. 2. Integration of SAS into CANSOC

SAS as an independent PC will facilitate its upgrade and maintenance. Furthermore, since it will be used at an end point it will not create interference with other equipment therefore ensuring the inclusion of other mission simulators in the future.

Conclusion

The use of the simulator is critical in mitigating any mission level risk. During any anomaly situation the operations teams' only line of defence against a mission failure will be the SAS. The simulator is also an effective tool to ensure that commands or command sequences that are detrimental to the spacecraft are not up-linked accidentally. The best argument for the need of a simulator is for those unknown scenarios that are not tested before launch. However, there has to be a trade-off between acceptable mission risk and cost to maintain the SCISAT-1 program within the realm of feasibility.

This paper outlined reasons to justify the selection of EuroSim and Mathworks products to develop a satellite operations simulator. The major drivers during this process were cost and development time. These goals can be achieved by the reuse of existing code, the simplification of the development process by the use of a graphical modelling environment, facilitating the transfer of models from Simulink to EuroSim using MOSAIC, easy integration of model code in EuroSim, and the simplification of the software maintenance process by incorporating reuse of simulator code by analysts and system experts in the SOC (using Simulink).

Maintenance of simulator hardware will be simple and inexpensive. By running on a PC connected to a standard network, most changes and upgrade can be done by any system administrator with absolutely no effect on other CANSOC components. The subsystem models can be developed individually and the level of fidelity can be changed at any point in the development process in order to respect project constraints or to increase functionality.

Acknowledgments

The authors wish to acknowledge the work and contributions of Bristol Aerospace and the Canadian Space Agency's Space Science and Satellite Operations branches.

References

[1] Walkty, I., Eliuk, W., Whitehead, B., Peterson, J., Doherty, T., Rumbold, G., Shelly, R., *SCISAT-1 ACE Mission*, Proceedings of the 11th Conference on Astronautics, CASI, Nov 7-9 2000.

[2] *SCISAT-1 Ace Simulator Requirements Specification*, SCICSA-SP0010-SAS-RS, Canadian Space Agency (CSA).

[3] Brunet, C., Foundy, K., Tafazoli, S., Hartman, L., and Adourian, C., *On designing the CSA SCISAT-1 ACE Ground Segment* DASIA, May 2000.

[4] Showalter, D., Parashar, S., Carpentier, L., Fortin, M., Hogan, P., *CANSOC: The Canadian Satellite Operation Center*, Canadian Space Agency (CSA), Proceedings of the 11th Conference on Astronautics, CASI, Nov 7-9 2000.

[5] Labezin, C., D'Hoine S., Mathieu E., *Multi-Mission Operation Centre* Study, Study done for the Canadian Space Agency, 2001, CS Communications and Systems Canada Inc.