RAMSES - A modern and flexible checkout and operational ground system for small satellite projects

Milan Battelino[1] and Christian Svärd[2]
OHB Sweden AB, Solna, SE-171 22, Sweden

RAMSES (Rocket and Multi Satellite EMCS Software) was developed in-house at OHB Sweden (former Space Systems Division at SSC) and has during the past years served a number of different space related projects; e.g. the PRISMA satellite formation flying project at OHB Sweden and DLR GSOC in Germany, the Russian scientific satellite project FOTON-M3 and the sounding rocket programs MASER and MAXUS. Common to all projects is that the RAMSES ground system is used throughout the whole project from development and test up to and including its mission phase. RAMSES is especially suited for projects with short timelines and where cost efficiency is an important driver. The system is deployed within minutes and executes on standard PC hardware. The adaptation of the system to new missions is generally only a matter of populating the system database with mission specific telemetry and telecommand definitions. The functionality within RAMSES is distributed between a number of standalone application nodes executing on one or several PC machines connected to the network. This paper will focus on the design, functionality and advantages of RAMSES compared with other systems on the market. Newly developed functionality concerning archiving and extraction of large amount of data is described. The role of RAMSES through the different phases of the small satellite project PRISMA is also presented.

I. Introduction

RAMSES[1] (Rocket And Multi-Satellite EMCS Software) is a general mission control system that is based on more than 30 years' experience in the space industry. The system is developed and maintained by OHB Sweden with new functionality constantly being added as new opportunities and areas of use arise.

The RAMSES system has been successfully used at SSC (Swedish Space Corporation), OHB Sweden and German Aerospace Center (DLR) in missions funded by ESA and Swedish National Space Board (SNSB) such as FOTON-M3, a series of sounding rocket campaigns (MASER and MAXUS) and the still active multi-satellite project PRISMA. New experiments are still being planned for and performed on the PRISMA platform where RAMSES plays a crucial role in the preparation, simulation and execution of each experiment.

The strength of RAMSES lies in its ability to be used across several phases in a space project: during testing and verification of software and hardware functionality, during development and integration of spacecraft prior to launch as well as to control and monitor spacecraft after launch. The early introduction of the same control system, that eventually will be used in operational phase, is a cost-effective de-risking strategy that cannot be underestimated.

The core of RAMSES consists of several GUI oriented application nodes. The nodes run on a dedicated LAN sharing the network traffic through IP multicasting, see Figure 1. This facilitates an open and scalable network architecture that gives a flexible and a cost-

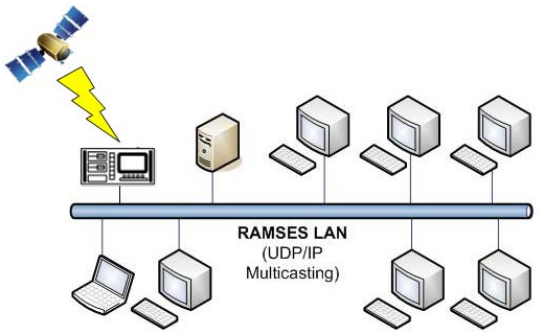

Figure 1. RAMSES network

[1] Software Specialist, AOCS & Software Department, P.O. Box 1064, SE-171 22 Solna, Sweden. Email: milan.battelino@ohb-sweden.se

[2] Software System Engineer, AOCS & Software Department, P.O. Box 1064, SE-171 22 Solna, Sweden.

effective system easily tailored for different projects as well as different phases within the same project. The RAMSES system is straightforward to configure and provides for a fast and straightforward setup. The core of RAMSES offers complete telemetry (TM) and telecommand (TC) functionality whereupon customized applications can easily be added and integrated providing project specific functionality. Support for the Consultative Committee for Space Data Systems (CCSDS) and European Cooperation for Space Standardization (ECSS) standards are built into RAMSES. The system is however not limited to these standards; through a straightforward adaption to the proprietary RAMSES protocol, the system may actually be used in a wide range of space missions and by virtually any space organization.

The core of the RAMSES system includes the following functionality:

- **Monitoring** function, through which telemetry and telecommand data is subjected to a range of monitor checks, where telemetry data is extracted, calibrated and may be further processed through the ECSS standardized Synthetic Parameter Expression Language (SPEL) and finally displayed
- **Commanding** function that prepares control messages, validates them and send them on for transmission
- **Control Procedure Execution** function that automates the execution of test, verification and flight procedures
- **Performance Evaluation** function used to evaluate spacecraft performance, in form of real-time graphs and telemetry displays
- **Time Correlation** function that maps the spacecraft elapsed time to UTC and vice versa
- **Data Distribution** function that gives the ability to service external requests for data being distributed on-line or off-line
- **Data Archive** function that implements the creation, management and maintenance of the mission archive, for access internally by other control system elements or for external access through data distribution
- **Operational Database** function that consists of a repository for all data (spacecraft, ground segment) defining the mission specific characteristics of the element subjected to mission control system processing functions
- **Supervision** function that monitors the status of the system, the traffic on the network and all executing applications in both real-time and by managing mission archive contents

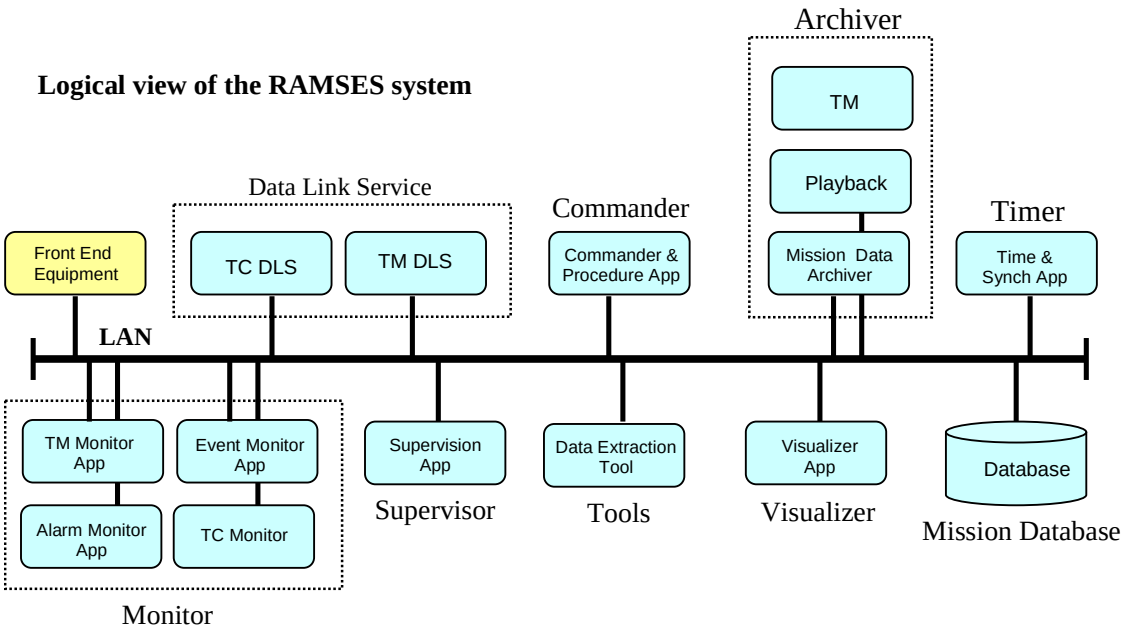

Figure 2. Logical view of the Ramses system

II. Architecture and Design

The RAMSES system is based upon a collection of inter-connected application nodes in an open network architecture where an extended UDP multicasting protocol is used, see Figure 2. Each application node is easily

"plugged" into the system and can immediately start receiving and processing data from other nodes, as well as sending data and communicating with other applications on the same RAMSES network. The implementation of a more traditional server-client solution was abstained for a number of reasons: such a solution has a potential inherent single-point-of-failure and is also less flexible through its server-centric architecture with a node that additionally has the risk of over time becoming feature-ridden and monolithic. Most of these pitfalls may of course be avoided through careful design, but the inherent flexibility and transparency of the open network architecture made this solution the preferable one.

A proprietary and versatile UDP multicasting based RAMSES protocol implements the distribution of vital information across the network between separate application nodes, each one with its own functionality and required data content. In the case RAMSES is used in a mission that adheres to relevant CCSDS standards, by encapsulating received CCSDS Transfer Frames[2] as well as extracted CCSDS Space Packets[3] into packets of proprietary RAMSES format, all applications connected to the multicast network are able to receive the available information carried by these packets. The proprietary format simply consists of additional headers before the CCSDS Transfer Frame and Space Packet content, as shown in Figure 3. For all types of encapsulated packets, here referring to Telemetry as well as Telecommand Transfer Frames[4] and Space Packets, a RAMSES Header is used. The RAMSES Header carries the length of the total packet, a timestamp, a flag that in case of Telecommand Frames and Space Packets indicates whether reliable transmission is used or not, and a timestamp. When the front-end receives a CCSDS Telemetry Frame, its main task is to add such a RAMSES Header and make sure that this RAMSES Telemetry Transfer Frame is multicasted to the RAMSES LAN. When the Telemetry Datalink Service application in the RAMSES network receives this packet, further extraction is performed: CCSDS Space Packets are extracted, a RAMSES Space Packet Header is created and inserted prior to the Space Packet content, the RAMSES Header received in the RAMSES Telemetry Transfer frame is updated (timestamp is kept intact) and used to encapsulate the RAMSES Space Packet Header and the extracted CCSDS Space Packet Data before being multicasted out on the RAMSES LAN.

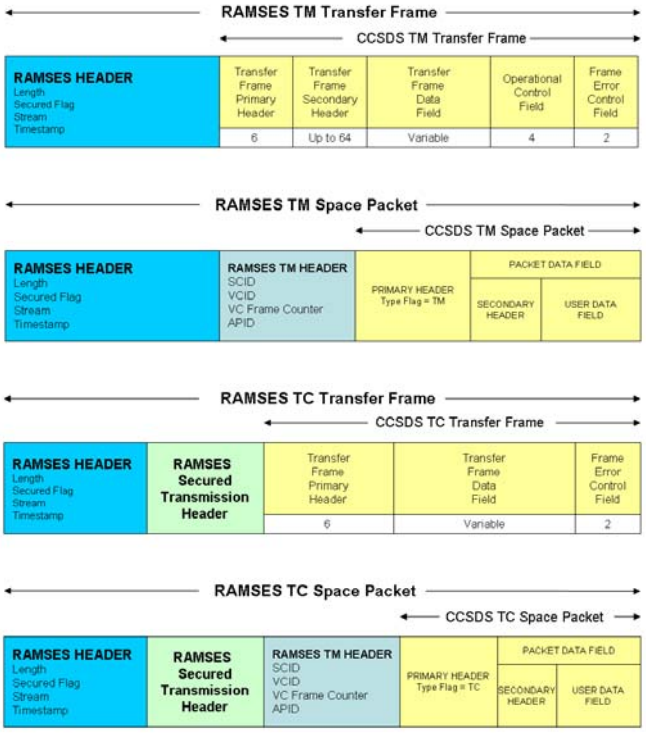

Figure 3. RAMSES TM and TC Headers and content

Each type of data in the RASMES network has its own multicast address and port range. In the example shown in Figure 4, the RAMSES TM Transfer Frames has a multicast address and port range from 239.255.3.0:55300 through 239.255.3.99:55399. A concept of "streams" is hence introduced through the use of a range of addresses and ports: by selecting a certain stream associated to a specific Spacecraft ID, it is possible to receive and process data from several spacecraft or other systems controlled on the same RAMSES network. This also facilitates playback of archived telemetry data without interfering with telemetry that is received in real time, by temporarily selecting another stream on the RAMSES network. Figure 4 shows an example of how application nodes with different functionality send respectively receives packets and frames made available through multicasting.

Since multicasting is performed using the connectionless and hence unreliable UDP protocol some means of ensuring reliable transmission of telecommands from commander through data links services to front-end is implemented. An additional "Secured RAMSES Header" to the RAMSES Telecommand Transfer Frames and RAMSES Telecommand Space Packets is in this case used together with a handshaking procedure between the sending and receiving node application for each individual Space Packet or Transfer Frame.

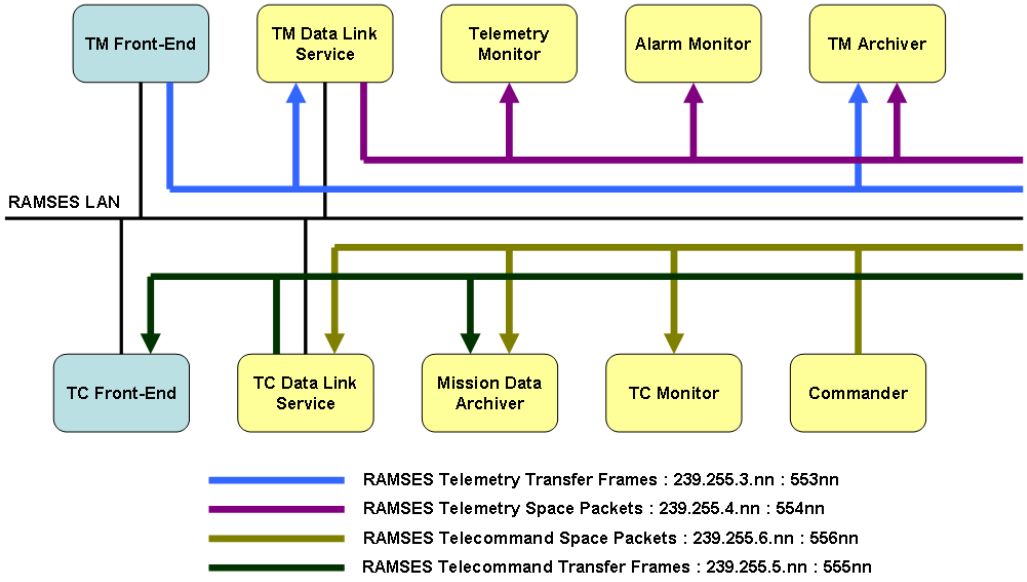

Figure 4. RAMSES Multicast Network

Each application node and software library component in the RAMSES framework is implemented with object oriented design methods and programming languages. Reusable software libraries in form of C# assemblies and native libraries written in C++ implement common functionality. These software libraries made it possible to quickly implement new user nodes and functions in RAMSES, adding to the scalability, transparency and flexibility of the system.

Well established object oriented design patterns were used throughout the design and implementation to ensure robust application with long term sustainability and an inherent scalability of the code. Standard PC with Windows was selected as platform for RAMSES the application nodes, which makes it possible to install RAMSES on any standard office PC. The selection of C# .NET for the GUI part of the applications and the "engine" code in C++ for the more time critical tasks ensured user-friendly MMI without any loss of performance, as may be expected from applications based solely on languages such as Java or C# .NET. A middle layer written in C++/CLI was required to ensure the interworking between the managed (C#) and native (C++) code, see Figure 5.

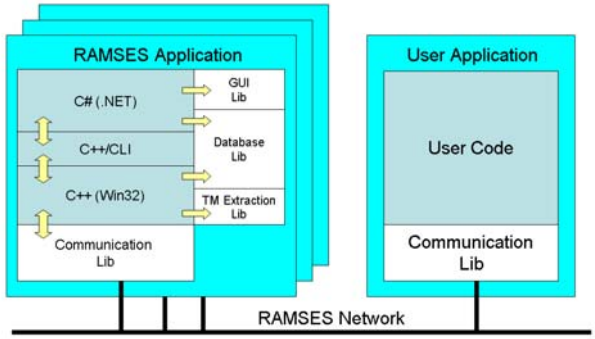

Figure 5. RAMSES Application Node Software Architecture

III. Functionality in Application Nodes

RAMSES does today consist of 16 different node applications, each covering separate functional areas. Each application node is named after an ancient Egyptian god or ruler.

Command Processing Engine - **Hathor** – provides the functionality, in which Data Link Service related control messages (commands) are prepared, validated, sent for transmission and verified according to CCSDS recommendations on the uplink from ground to the space segment.

Telemetry Processing Engine - **Nefertiti** - the Data Link Service that process data transmitted from space segment according to CCSDS recommendations on the downlink to ground. Nefertiti also produces statistics for further analysis based on processed data.

Telemetry Displayer - **Sphinx** – provides the functionality in which all data, regardless of source, can be extracted, calibrated, subjected to a range of monitor checks and displayed. Sphinx also has a built in Alarm Monitor that makes it possible to supervise the alarm status of any telemetry parameter with an associated range set.

Control Procedure Executor - **Cheops** – provides the functionality to automate the execution of flight procedures with PLUTO, a procedure language used for test and operations defined by ECSS. Cheops also provides a procedure development and execution environment.

Telemetry Saver - **Anubis** - provides the functionality to create, manage, and maintain the mission archive containing telemetry data, for access internally by other control system elements or for external access through data distribution.

Telemetry Playback - **Osiris** – provides the functionality to distribute data offline. Recorded real-time data can be replayed at any required speed.

Telecommand Status Monitor - **Chephren** – provides the ability to monitor the transmission and execution status of Telecommand packets, and of scheduled Telecommands issued through the on-board operations scheduling service, an implementation of the ECSS PUS Service Type 11.

System Synchronizer - **Ankh** – provides the functionality to synchronize the system time either with a simulated time or real time and also provides the functionality to map the spacecraft elapsed time to UTC and vice versa.

System Supervisor - **Obelisk** – provides the functionality to monitor the status of the system, the traffic on the network and all executing applications in both real-time and by managing mission archive contents.

Telemetry Extractor - **Nut** – provides the functionality to export extracted and calibrated data to external evaluation tools such as Matlab® and STK® (Satellite Tool Kit).

System Saver - **Toth** – provides the functionality to store all data (both broadcast and private) transmitted on the RAMSES network, except for telemetry data, where Toth only store if telemetry data is received or not.

Alarm Monitor - **Aker** – provides the ability to perform range checks of telemetry data (calibrated values as well as raw values), to monitor storage disks and to generate alarms as soon as monitored values are outside given limits.

Event Monitor - **Seth** – provides the functionality to monitor Telemetry source packets relating to ECSS PUS Event reporting Service.

Data Visualizer - **Maat** – provides three-dimensional STK based visualization of spacecraft attitude, position and vector orientation in real-time. Maat can be used to monitor any kind or any combination of spacecraft in a mission.

Telemetry Database Server – **Ra** - implements the ability to extract and store a large amount of telemetry parameter data in an indexed SQLite database. The user gets near instant access to the content or trends of any subset of parameters from any selected time period covered by the telemetry dataset. The on-board time (OBT), same as Spacecraft Elapsed Time (SCET) or Earth Reception Time (ERT) is used to define the timespan. The TM parameters may then be further filtered using e.g. VCID (Virtual Channel Identifier) or APID (Application Process Identifier), see Figure 6.

RAMSES Mission Database - **Pyramid** - is interfaced by the majority of the node applications and contains the description and definition of mission specific commands, telemetry parameters and packets, synthetic parameters, calibration curves for data as well as range sets. Pyramid supports CCSDS format packets and ECSS PUS data formats. Pyramid is a Microsoft® Office Access™ database and does support export/import to/from the SCOS MIB format.

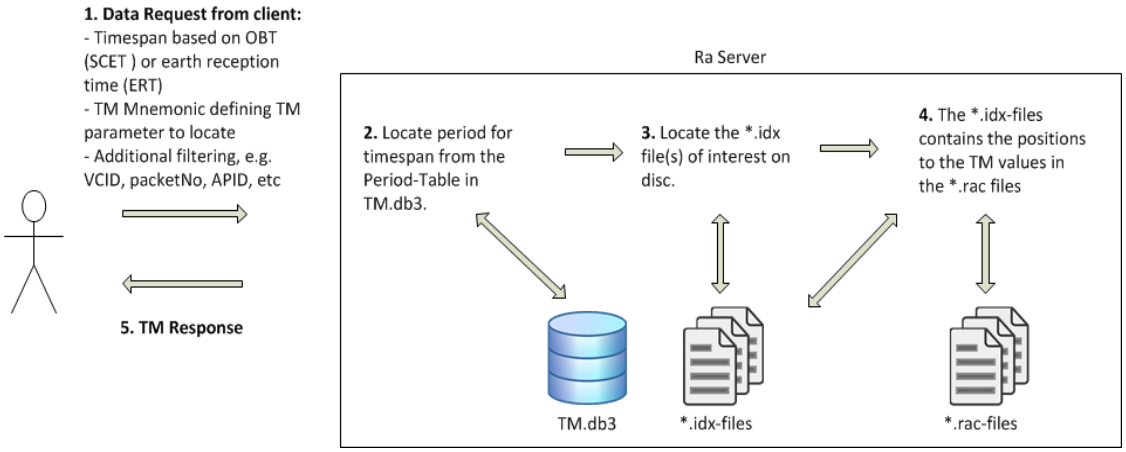

Figure 6. Ra telemetry data request overview

IV. RAMSES in PRISMA, a small multi-satellite mission

PRISMA[5] is an in-orbit multi-satellite experiment consisting of two spacecraft flying in formation: MANGO and TANGO. The mission is funded by the Swedish National Space Board (SNSB) with in-kind contributions from German Aerospace Center (DLR), Technical University of Denmark (DTU), and French Space Agency (CNES). OHB Sweden AB is the prime contractor for the mission.

The MANGO spacecraft has full three-axis attitude control capability as well as full three-dimensional attitude independent orbit control capability, and is equipped with two experimental propulsion systems. TANGO, with no active orbit control capability, implements a magnetic attitude control with determination supported only by sun sensors and magnetometer providing three-axis stabilization. Experiments in autonomous formation flying and rendezvous are achieved through new technologies in form of hardware as well as algorithms and strategies[6]. Three different sensor systems are used during the experimental flights: a GPS navigation system from DLR, the Vision Based Sensor from DTU, and the Formation Flying RF-sensor (FFRF) from CNES. DLR and CNES have also developed guidance and control functions that have been incorporated into the onboard software.

A. SATSIM, a real-time satellite simulator

A real-time satellite environment simulator, SATSIM[7,8], was developed within the PRISMA project to handle and support the different phases. This simulator is fully configurable through the RAMSES system. SATSIM and RAMSES are used to initialize different simulation scenarios, modify parameter values or to inject errors during simulation in sensor or actuator models and is also used during operations to test and verify different experimental flight scenarios. SATSIM did together with RAMSES implement SATLAB, a real-time simulation and test system that also included the onboard computers. This setup was implemented early within the PRISMA project and was primarily used for software development and system tests of the onboard software. As can be seen in Figure 7 SATSIM, simulating both satellite hardware and environment, is connected to the onboard computers through CAN buses as well as to the RAMSES network.

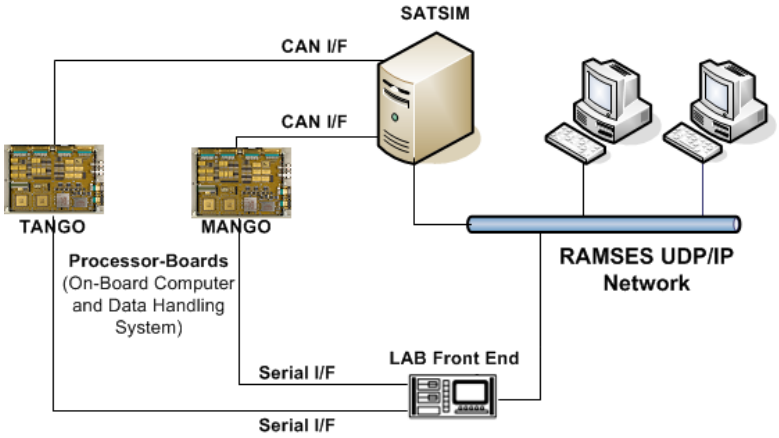

Figure 7. High-level view of the SATLAB environment

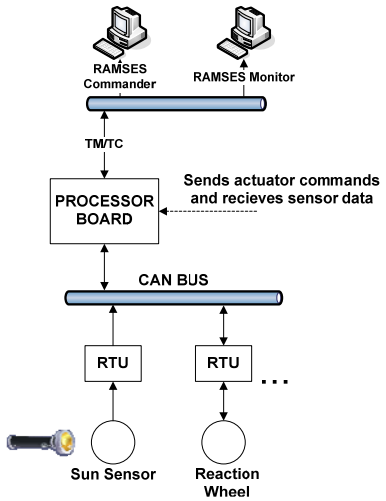

Figure 8. Open-loop test setup

B. Hardware integration tests

In a later phase, during integration of hardware in the two PRISMA spacecraft, open-loop tests were performed using RAMSES, without SATSIM, to verify the functionality of the hardware. Different sensors were stimulated with well-defined input, and the output was then analyzed. Figure 8 gives a simplified view of this type of setup that was used in the PRISMA project and shows an example of when the sun sensor is illuminated with a known input. The sensor outputs data via an RTU (Remote Terminal Unit) reaching the processor board through the CAN-interface. In the RAMSES system TM is monitored through e.g. graphs to verify the functionality. When verifying the functionality of e.g. the reaction wheel, the RAMSES command processing engine is sending TC-commands that reach the on-board processor board via a front-end, not shown in Fig. 18. The processor board then commands the reaction wheel to perform a change in e.g. velocity. The output from the reaction wheel is then analysed in the RAMSES monitor. In conjunction with the sensor tests the sign of the output is also verified, especially against the

on-board GNC-system software. Another test that is performed during the integration test phase is to verify that current and voltage does not exceed limit values that can be harmful for the spacecraft hardware.

C. System functional and performance tests

During System Functional- and Performance Tests, SFPT, closed-loop tests were performed where the functions of all HW systems and its interaction with the onboard SW were exercised on the PRISMA satellites. A more complex test setup was required that incorporated both SATSIM and RAMSES. The design of SATSIM did facilitate the possibility to have a mix of real and simulated Remote Terminal Units, a useful option since hardware availability during some of the tests was limited. PUSIMs (Peripheral sensor Unit Simulators) did represent in this test setup the real sensor and actuator hardware to which SATSIM was physically connected. SATSIM did during this tests only listen to CAN data traffic. PUSIMs were instructed via commands sent by SATSIM over the RAMSES network, based on simulator dynamics and commands received from the RAMSES commander node, see Figure 9.

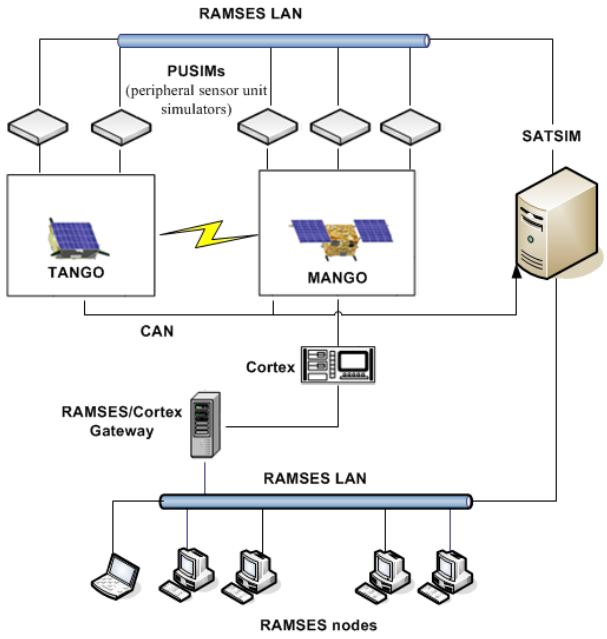

Figure 9. Test setup during the SFPT test campaign in PRISMA

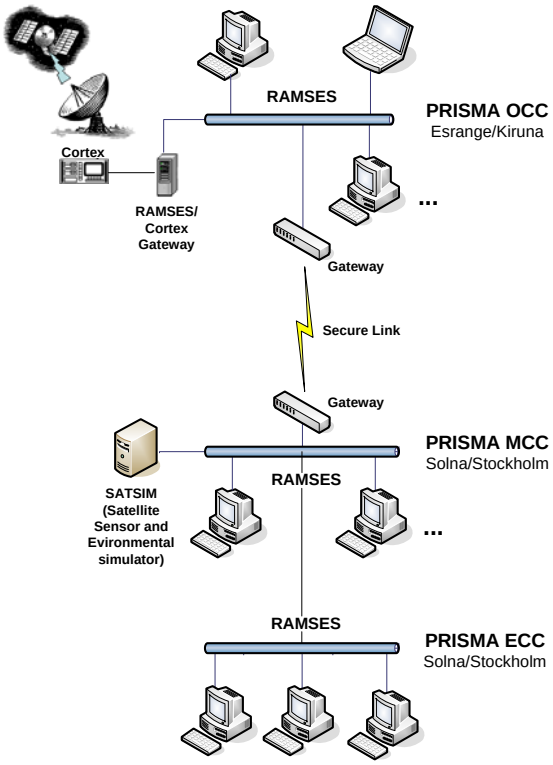

Figure 10. High-level view of the PRISMA operations setup

D. Operations

On June the 15th 2010 at 14:42:16 UTC the two PRISMA spacecraft were launched, clamped together, by a Dnepr rocket from Yasny, Russia, into an sun-synchronous orbit. The RAMSES system was from that point of time used at different the control centers. PRISMA operation is divided into different control centers housed at different geographically locations in Sweden. The main ones are the Operations Control Centre OCC at Esrange in Kiruna, Mission Control Centre MCC in Solna and the Experiment Control Centers ECC also in Solna.

The PRISMA Operations Control Center, OCC, has as its main task to perform the actual upload of commands defining the experiments onboard the PRISMA satellites and also to monitor health status of the satellites. In the case of a failure either in hardware or in the onboard software the OCC has the authority to abort an ongoing experiment. The control system used at the OCC consists of several inherent functions, such as RAMSES as EMCS system, a GNC viewer software dedicated to analyze GNC performance, an antenna controlling software named Oasis and a passage planning tool.

The PRISMA Mission Control Centre, MCC, is located in the main facilities of OHB Sweden in Solna. The main function of MCC is the upload of experiments onboard the PRISMA satellites, the monitoring of the PRISMA hardware and software health status against predefined limit values and also

planning, scheduling and the execution of the timeline of the mission. The MCC also has as objective to monitor and validate the incoming telemetry data ensuring the safety of the satellites. MCC has the authority to, based on this information, reject an ongoing experiment and reschedule the time line of the mission if required. Several different software and hardware modules are utilized at the MCC, including RAMSES, SATSIM, dedicated flight dynamics and mission planning software, and same GNC viewer software that is used in the PRISMA OCC. SATSIM is used at MCC to simulate and verify an experiment before uploading it via the OCC.

The PRISMA satellites are hence commanded from either the OCC at the SSC facilities in Esrange in Kiruna or from MCC at the OHB Sweden facilities in Solna. Figure 10 gives a schematic view of the different centers and their interconnectivity.

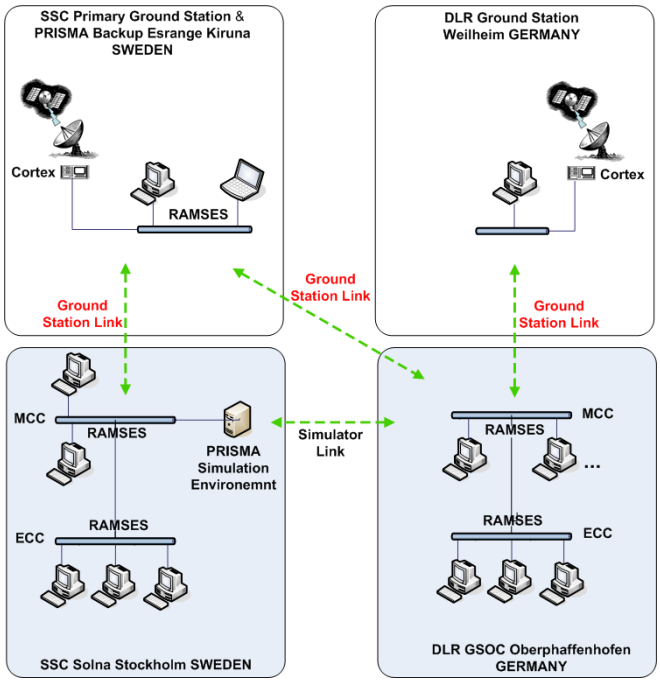

Figure 11. Simplified view of the operational setup with DLR GSOC

In mid-march 2011 the operational control of the PRISMA satellites were, for a period of 5 months, handed over to German Aerospace Center (DLR) at the German Space Operations Center (GSOC). A duplicate RAMSES environment was set up at the GSOC premises in Munich, as shown in Figure 11. The setup and handover of the operational control to GSOC was smooth and successful, demonstrating the flexibility and reliability of the RAMSES system.

Until this date the PRISMA mission has resulted in a huge set of telemetry data stored in RAMSES archive files. Such a large dataset does require a more user-friendly and efficient access than the simple playback of archive files with telemetry data that RAMSES so far did offer. The solution was to implement a new RAMSES node, Ra; an addition to the data archiving functionality. The main task of Ra is to autonomously extract all telemetry data from RAMSES archive files, store it into an index-based relational database and give the user near instant access to any set of PRISMA parameter data across any selected time period covered by the PRISMA mission. The data is through Ra filtered and presented either alphanumerically or graphically, as trends or calibrated parameter values. Needless to say, this latest addition to the family of RAMSES nodes is much appreciated by PRISMA experimenters and mission managers.

V. Conclusion

RAMSES has in several projects been proven to be easy to configure, install and deploy. Since RAMSES not only conforms to CCSDS and ECSS standards, but is easily adaptable to other protocol formats, it is usable in wide range of different missions, environments, applications and roles. An example described in this paper is the PRISMA mission that demonstrates how RAMSES is used through the different phases in a multi-satellite project and the inherent advantages of having such a system. By using RAMSES as control system from early engineering and development through all different testing and validation stages up to launch, a completely validated and reliable mission control system and mission database is ready for the operational phase. The new telemetry database functionality also ensures that experimenters and mission managers have near instant access to any subset of the house-keeping and payload data collected over a complete mission.

The network centric nature of RAMSES gives it a series of advantages compared to similar systems on the market, such as scalability, transparency, that it is easy expandable and has a built-in fault-tolerance. The scalability of RAMSES makes it easy to adapt to needs of different applications of control systems; a freely selectable subset of RAMSES application nodes can be installed on various configurations, on one PC or several, in one network or more. All data received and processed is transparently available through the RAMSES network; dedicated application nodes can easily access, process and present data. New mission specific user nodes are easily

implemented on the Windows or Linux platform through the RAMSES communication library component. If the control system application requires high fault-tolerance for one or several system functions, it is just a matter of installing extra duplicated RAMSES application nodes on additional PCs, plugging in them on the RAMSES network and in this way avoiding single point of failures.
For OHB Sweden and Swedish Space Corporation this is a cost-effective solution, since the RAMSES system can, with small effort, be re-used between different satellite missions and rocket campaigns. The effort generally lays in updating the mission database and in some cases the development of mission specific applications with the support from the existing RAMSES software libraries.

Acknowledgments

On behalf of OHB Sweden AB, the authors would like to thank the Swedish National Space Board (SNSB) for having funded the further development of RAMSES. The PRISMA project was funded by the SNSB with in-kind contributions from German Aerospace Center (DLR), the Technical University of Denmark (DTU), and the French Space Agency (CNES).

References

[1]A. Carlsson et al, "RAMSES – A General Control System for both Sounding Rockets and Satellites", *18th ESA Symposium on European Rocket and Balloon Programmes and Related Research*, Visby, Sweden, 2007, pp. 181-186

[2]CCSDS 133.0-B-1: "Space Packet Protocol", September 2003

[3]CCSDS 133.0-B-2: "Packet Telemetry Service", June 2001

[4]CCSDS 232.0-B-1: "TC Space Data Link Protocol", September 2003

[5]S. Persson, S. Veldman and P. Bodin, "PRISMA – A Formation Flying Project in Implementation Phase", *58th International Astronautical Congress*, IAC-07-B4.2.09, Hyderabad, 2007

[6]S. D'Amico, S. De Florio, R. Larsson and M. Nylund., "Autonomous Formation Keeping and Reconfiguration for Remote Sensing Spacecraft", 2009, 21st ISSFD, Toulouse, France, 2009

[7]S. Persson, "The Validation and Testing of the PRISMA Formation Flying Project", *59th International Astronautical Congress*, 2008, IAC-08-B4.2.2, Glasgow, Great Britain

[8]P. Bodin, M. Nylund and M. Battelino, "SATSIM – A Real-Time Multi-Satellite Simulator for Test and Validation in Formation Flying Projects", *Acta Astronautica 74*, 2012, pp. 29–39