The ESAIL Multipurpose Simulator

Workshop on Simulation and EGSE for Space Programmes (SESP)
26 - 28 March 2019

ESA-ESTEC, Noordwijk, The Netherlands

Yago Isasi Parache [(1)], Aleix Pinardell[(1)], Antonio Márquez[(1)] , Christophe Molon-Noblot [(1)],
Alexander Wagner[(1)], Marc Gales[(1)] and Miroslav Brada[(1)]

[(1)]Luxspace Sàrl
9, rue Pierre Werner L-6832 Betzdorf
Grand Duchy of Luxembourg
Email:isasi@luxspace.lu

INTRODUCTION

ESAIL is a microsatellite developed in a PPP together with ESA and ExactEarth in Luxembourg. The satellite payload is an AIS Receiver for vessel-detection from space. During the development of the satellite, a multipurpose simulator was created to support several activities of the satellite development during its phases C and D. Taking in consideration the different facilities described in ECSS-E-TM-10-21, the ESAIL multipurpose simulator has been used in different configurations to produce the different facilities required by the ESAIL project. During the development of such simulator facilities and for the creation of simulation models, different technologies have been used. Those are described along this paper. In the first sections, we focus on the relevant aspects of the ESAIL satellite system design and the derived simulation models' characteristics and fidelity. The later section describes the different facilities used during the ESAIL project, their purpose within the project lifecycle and their schematic architecture and building blocks.

THE ESAIL SATELLITE

The mission of ESAIL is to collect AIS signals from vessels all over the globe and provide the collected data to ground. The data is either to be processed on board (OBP) or on ground (OGP) depending on the purpose such data must serve. Therefore, the S/C can run in OBP and OGP both individually and simultaneously.

ESAIL system design

The satellite design is targeted to reach the following objectives: high reliability, high level of autonomy in nominal mission and low cost. These objectives are implemented with a system composed of the following functional chains:

- On-Board Data Handling: responsible for controlling the spacecraft (platform and payload), managing the operative modes, collecting and storing housekeeping data.
- Electric Power System: responsible for collecting, storing and distributing electrical power.
- Telemetry & Telecommand: responsible for receiving and routing platform and payload telecommands and for downloading housekeeping data to ground.
- Attitude determination & control: responsible for estimating and controlling the attitude of the satellite.
- Payload: responsible for collecting, storing and downloading AIS data to ground.
- Structure: responsible for protecting the system from external loads and 1for the deployment of platform and payload antennas.
- Thermal Control: responsible for ensuring that each component of the spacecraft remains within their operational temperature ranges.

The satellite implements the following redundancies:

- OBC (2 cold redundant units)
- TMTC (2 hot redundant receivers, and 2 cold redundant transmitters, sharing antennas and signal routing)
- ADCS
    - Software on two redundant OBCs.
    - 2 cold redundant ADCS I/F units.
    - 2 cold redundant gyros.
    - 2 cold redundant magnetometers.
    - 4 reaction wheels in 3 orthogonal + 1 diagonal configurations (3 units at a time will be ON)

    - 3 orthogonal magnetorquers, each with internal redundancy
    - 6 fine sun sensors, 3 on each ADCS I/F unit.
- Payload subsystem:
    - Payload Data Downlink (2 cold-redundant chains, sharing a single switch and single antenna)
    - AIS Receiver (with internal redundancy, 2 cold-redundant chains)
- The EPS is not redundant, but certain elements such as the PCDU FPGA, are present in cold redundancy

The Payload Data Handling Unit (PDHU) is not redundant.

INITIAL STEPS AND SIMULATOR HIGH LEVEL REQUIREMENTS

LuxSpace had already accumulated experience on the development of simulators during the SGEO HAG1 and EDRS-C missions, however there was an initial concern of the real need to develop a simulator in the frame of a program such as ESAIL, having important constraints in terms of budget and time. The option of using hardware based facilities and testing platforms instead of a software based simulator was preliminary discussed. During these discussions there were a set of reflections that show important risks of basing a validation and verification strategy using hardware-only means: Typically, hardware availability has a schedule that depends on a certain number of supplier schedule constraints that are not always respected. Thus, linking by example the OBSW validation schedule to a hardware availability date may produce an automatic delay on the OBSW validation plan if the hardware is late. Another point is that the simulation of the hardware failures could be complex (or impossible) to generate and record when only hardware means are available. Moreover, the fact to avoid working with the real hardware, reduces the risk of producing undesirable damages on the hardware equipment. Even when a small part of a hardware parts fails, when a replacement is needed, if the spare part is not available, there is an important dependency on the supplier that could impact the schedules. Finally, the possibility of parallel development and testing is reduced when only hardware means are available. These preliminary discussions, and hardware schedule constraints, brought the need of developing a simulator that could fulfil a number of needs. For high-level requirements:

1. To support the software development and validation of the ADCS-IF board (hardware and software) as a subsystem
2. To support the validation of the ESAIL OBSW including:
    a. OBSW functional validation in open and closed-loop.
    b. OBSW integrated tests.
    c. SRDB OBSW relevant parameters validation.
    d. Failure injection simulation controlled by the simulator operator.
    e. Capacity to run either real-time or faster.
3. To allow debug of the OBSW debug being able to run step by step execution and to set breakpoints:
4. To allow the OBSW testing, facilitating that every single developer/tester/end user can run an instance of the required facility on an office PC.
5. To allow the execution of OBSW tests in an automatic fashion (e.g. by means of using a continuous integration tool like Jenkins).
6. To integrate with the AIT EGSE CCS tool (that provides the spacecraft database to support model initialisation and the TM/TC encoding/decoding) that will be used for the satellite system V&V activities.
7. To allow the execution of operational procedures for the operators of the satellite.
8. To support the testing of future OBSW updates as required during the maintenance phase.

ESAIL MULTIPURPOSE SIMULATOR INFRASTRUCTURE

The ESAIL Multipurpose simulator follows the general block diagram architecture of the virtual system model shown in Fig. 1. The used simulation infrastructure was **Rufos** (Runtime for Simulation). Rufos is the OHB's SMP2 compliant simulation runtime environment that hosts the simulation models of the simulator facility and provides necessary services to configure and control these models during a running simulation ([3][4][5]). Rufos implements the different simulator states and the mandatory simulation services as required by the **SMP2** simulation standard [6][7][8]. It also resolves the need to provide interfaces to control the simulation via a Man Machine Interface (MMI) for interactive simulations or via scripts for the execution of automatic procedures. Moreover, the Rufos "Platform Models" denote a set of SMP2 compliant models, which implement generic functionalities that recur within the different simulator facilities and could also be used in the context of ESAIL. In particular, the following software models were of interest for ESAIL

- The "Common Models" that are SMP2 compliant simulation models of standardized spacecraft equipment that is intended for usage in various missions. For ESAIL OHB developed a model of the GR712RC board.

- The “Line Models” implement simulation models of the standard electrical interfaces of spacecraft systems. In particular models such as SpaceWire (SpW), the CAN-BUS and several discrete signal lines were of interest for the ESAIL simulator.
- The GNU Debugger (GDB) Interface Model provides the connection to an external tool for debugging the OBSW.
- The Monitoring and Control (M&C) Interface Model simulates the communication interface between the spacecraft and the ground station. It is connected to an external M&C facility with which exchanges telecommands (TCs) and telemetries (TMs) using a pre-defined protocol.

The generic view of the ESAIL simulation facility architecture is shown in Fig. 1.

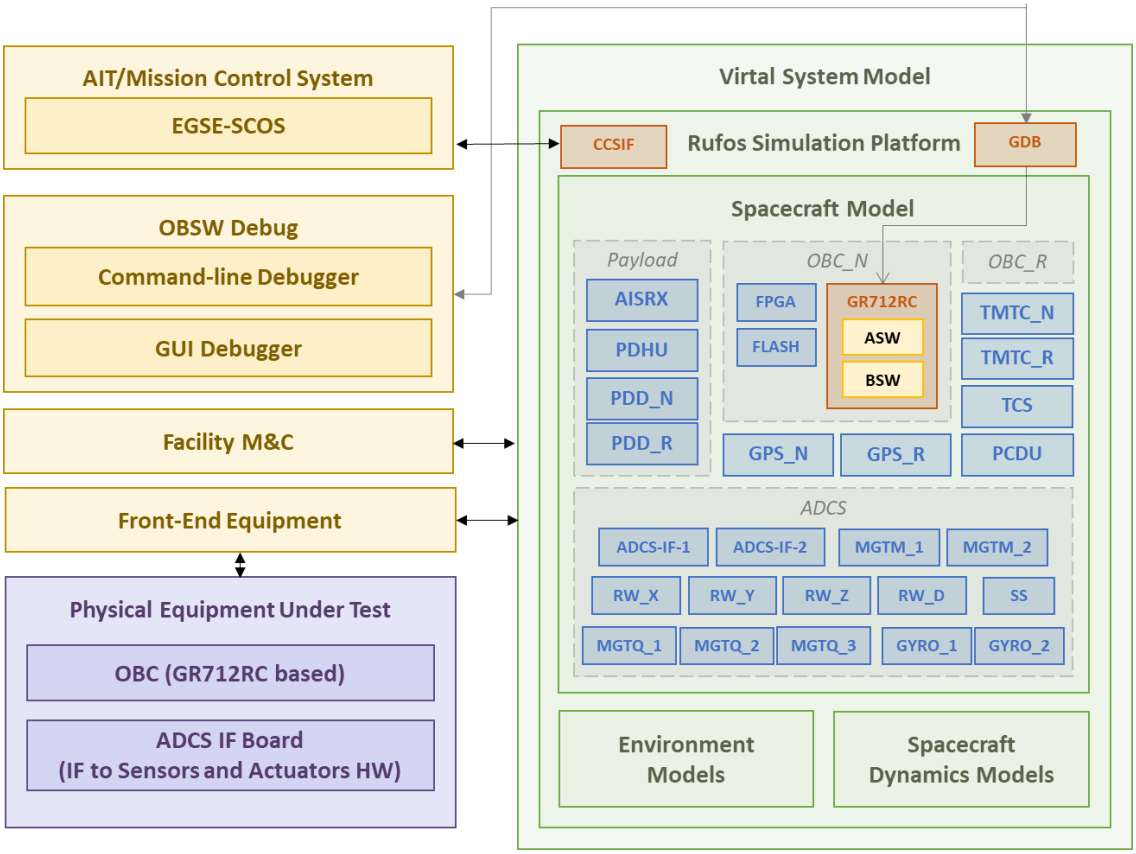

Fig. 1 ESAIL Simulation Facility Architecture

The different facilities presented across this paper can be derived from this generic schema by configuration of the proper assembly of composed elements.

Simulation Models

A rich set of SMP2 simulation models have been developed in the frame of the ESAIL multipurpose simulator. Some models where provided by the OHB platform models including:

- The CPU GR712RC common model.
- Line models such us the CAN and UART line models.
- GBD Model.
- CCSIF Model.

Other models were specific models developed by ESAIL:

- The OBC excluding the CPU (GR712RC) model (OBC Flash Memory and FPGA model).
- The GPS model (based on Phoenix-S receivers)
- The TM/TC board model.
- The payload models: PDHU and PDD models.
- The ADCS IF board model.

- The gyroscope model (based on the AstroFein AGS-1).
- The magnetometer model.
- The magnetorquer model.
- The reaction wheel model (based on the AstroFein RW90).
- The sun sensor model.
- The EPS model including the PCDU, LCLBoard, Battery, Solar Array and FPGA Models.
- The TCS subsystem including the heater line models.
- Common models developed by LXS that are not project specific, so that they can be reused across projects (e.g ADConverter, CANNode or PowerSource).

Model requirements were defined using DOORS and exported to MagicDraw using ReqIF. Models were designed using the pre-defined SMP2 stereotypes generating SMP2 compliant C++code and test skeletons directly from the models. These code and test skeletons were completed by the model developers within the Eclipse IDE. After model tests execution, test and test result information were automatically retro-feed into DOORS using DXL scripts.

OBDH Modelling

The brain of the simulator is the implementation of the emulated OBC of the ESAIL Satellite and more extensively its OBDH. The OBDH system consist of the following modules:

- OBC, consisting of
  - OBC (HW) including the FPGA and Memories
  - OBC SW (OBSW)
  - OBC FPGA firmware
- GPS

The OBDH has two functions:

- To manage and control the platform and the payload through the OBC and the OBSW
- To provide orbital position and onboard time knowledge through the GPS.

The general physical architecture of the OBDH functionality is shown in Fig 2,

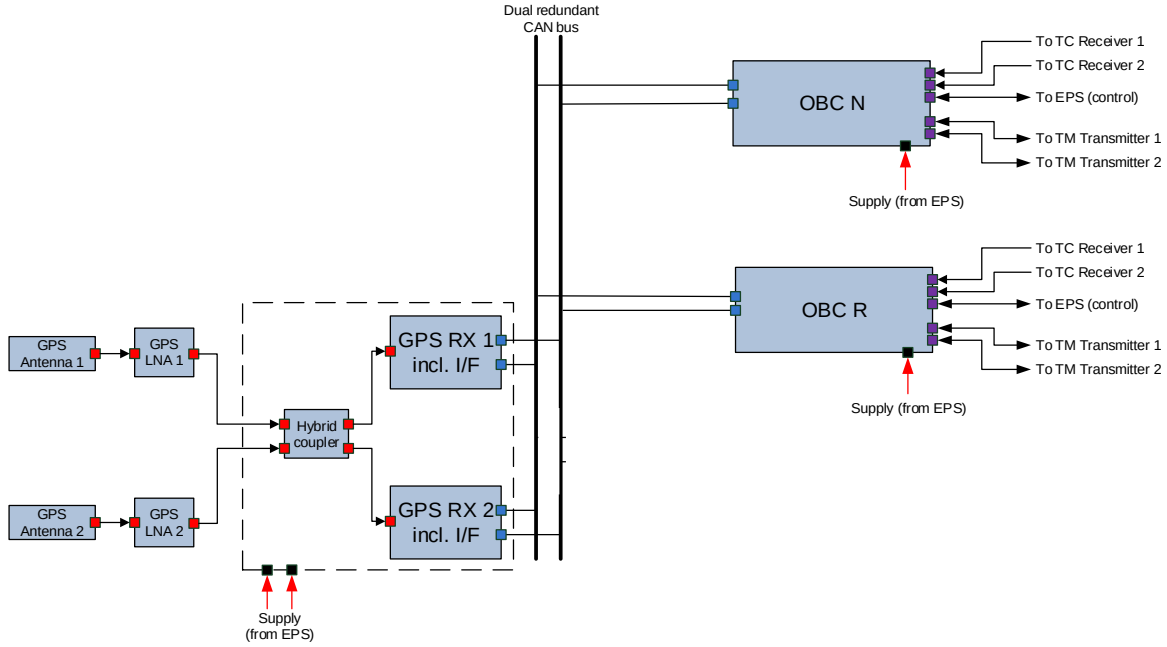

Fig 2 ESAIL OBDH Physical Architecture

The main parts of the OBDH physical architecture are the two redundant OBC (N: nominal, R: redundant). The OBC connects to the other equipment either by a direct link or through the dual redundant CAN bus. For the time reference, the OBC is supported by two GPS chains, which consists of antennas, LNA and GPS receivers. The OBC is a classical microprocessor system based on the space qualified GR712RC board that provides most of the required functionalities

and interfaces. Different type of memories are integrated into the design (SRAM, Boot memory, Flash mass memory, MRAM). The FPGA (Actel RTAX1000S-CG624) is used to provide the functions and interfaces that are not covered by the processor. The modelling approach was to have a medium fidelity representation of the OBDH subsystem, especially high on the OBC FPGA, Memories, interface modelling for the OBC and GPS whereas the fidelity needs were lowered on the antenna and coupler models. The redundancy is achieved by two instances of the same model for OBC and GPS.

Several processor interfaces were modelled such as the CAN, the CCSDS compatible GR712RC TC and TM interfaces, the singled ended processor GPIOs, the UART monitoring interface providing the sniffing capabilities to the OBSW and the Housekeeping ADC. For the CAN interfaces, dedicated CAN lines were modelled representing the CAN 2.0B controllers that are integrated in the GR712RC processor and the external CAN transceivers. Each controller/transceiver is also connected to one of the redundant CAN-BUS as in the real system. Elements connected to the CAN lines use dedicated CanNode components. A simplified schematic of the OBDH simulation models connected to the CAN-BUS is shown in Fig 3

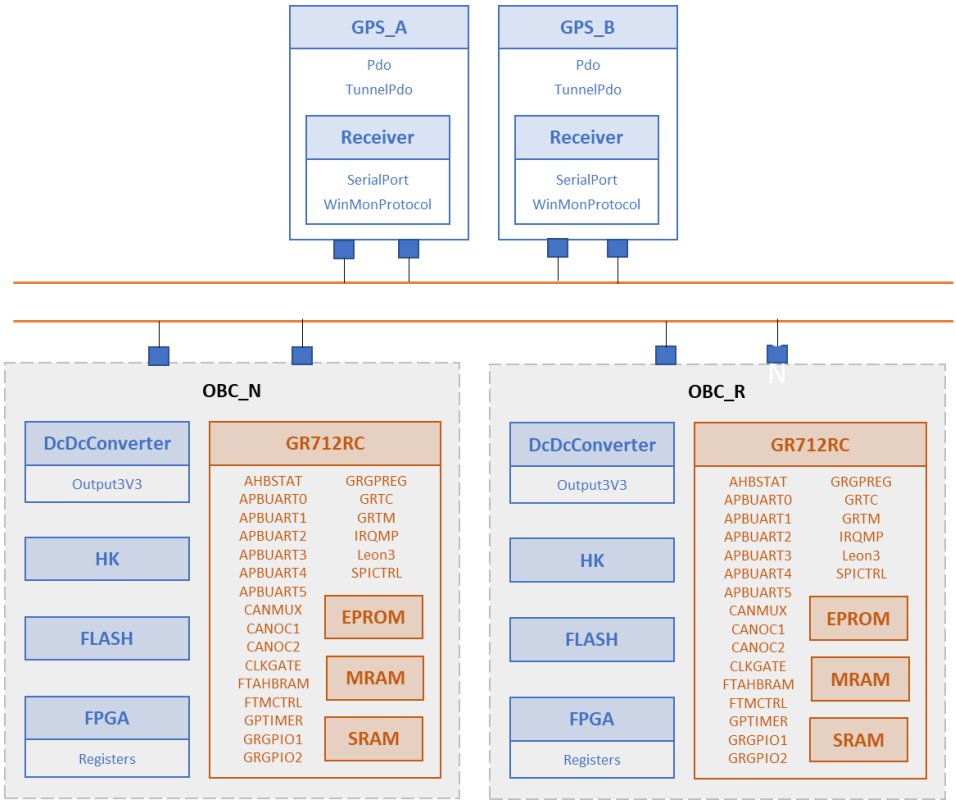

Fig 3 ESAIL OBDH subsystem model

EPS Modelling

The EPS subsystem has been modelled to allow realistic control of the spacecraft. As part of the EPS subsystem the following models have been developed:

- PCDU Model: including the LCLBoard, FPGA and Non-Volatile Memories have been modelled including the real redundancies.
- Battery Model.
- Solar Array Model.
- Generic Power Models: Relays, ADCs or LCS´s models.

The different equipment is connected to the EPS subsystem, and the power on and off of each equipment can be controlled by sensing the appropriate commands to the LCLs.

ADCS Modelling

The attitude determination and control subsystem has been modelled in two steps. In the first step the models were designed to allow debugging and validation of the OBSW, making more emphasis on the interface modelling compared to the accuracy of the equipment responses. On a second step the models were extended to allow more accurate responses close to the real system responses interacting with a dynamic model of the satellite environment with the objective to provide. The models of the sensors and actuators are connected to the ADCS IF model and provide information of their status to the board. Real redundancies of the system are also fully achieved by proper instantiation of the board, sensors and actuators SMP2/C++ classes.

PL Modelling

The models of the payload offer a reliable characterization of the different payload modes and the transitions between them. The payload modes and the status of the different payload models is summarized in Table 1

Table 1 ESAIL PL units´states for each payload mode

P/L Mode	PDHU	AIS RX	PDD
OM0	OFF	OFF	OFF
OM1	ON	ON	OFF
OM2	ON	ON	OFF
OM3	ON	ON	OFF
OM4	ON	OFF	ON
OM5	ON	ON	ON
OM6	ON	ON	ON
OM7	ON	ON	ON

For each PL equipment, the commands and the associated telemetries have also been modelled together with all the redundancies of the PL subsystem.

TCS Modelling

The thermal control subsystem modelling was one of the most critical models of the ESAIL simulator since there was no possibility to test the subsystem on other platforms. For that reason, the system and its models were implemented to allow a trustworthy simulation and reliable validation of the OBSW TCS implementation in open and closed loop.

ESAIL SIMULATION FACILITIES

Taking in consideration the initial needs and requirements, several facilities were built using the different models described in the previous sections. The following sections will describe the configuration of the different facilities that have been used within the ESAIL project.

Functional Validation Testbench (FVT)

The ESAIL satellite rely for its ADCS subsystem on the so called ADCS-IF avionic board, used to connect all Sensors and Actuators of the satellite. To support the analysis of system performance and the test and validation of the ADCS-IF (HW and the embedded SW developed by LuxSpace) a Functional Validation Testbench (FVT) was used. The scope of this simulator was in the ESAIL case not a complete system, but just a reduced configuration a with a focus on the identified critical bread-boarded and engineering model of the ADCS-IF hardware and software. This configuration consisted in a single OBC model, running a light OBSW containing the ADCF-IF Handler task. The ADCS-IF Control application (acting as facility M&C) was used to send direct commands to the OBSW ADCSIF-handler queue of the OBSW. These commands were later forwarded to the ADCS-IF board, that was connected to the simulator through a serial port of the computer used to run the overall simulator infrastructure.

Software Validation Facility (SVF-SW)

The simulator has been also used as SVF with essentially 4 main purposes. For one side to help the developers during the development of the OBSW, and later to perform formal validation of the OBSW. For the later, the risk of adaptations of model code to wrong OBSW behaviour was minimized by having different teams providing the OBSW and the SVF solutions. Another purpose was to introduce failure injection scenarios that cannot be tested easily by the available hardware. Develop procedures for OBSW validation that can be reused during the Flatsat campaign. The SVF was not be used to simulate processing and communication delays of the real HW. The ESAIL SVF follows the architecture of the virtual system model defined in [1] as SVF-SW where a processor model was used instead of the real

OBC hardware in the loop (PIL configuration). In the frame of ESAIL the current implementation of the facility configuration described in Fig 4 includes the on-board computer emulator and the equipment models. No Spacecraft Dynamics and Environment Models were required for the facility concept for ESAIL. One of the advantages of the SVF, is that provides an execution environment that can be controlled down to the level of a single processor cycle. It also can be started, stopped, single stepped, and context frozen at any point in the OBSW execution. Thanks to these features, it was possible to trace and monitor the software for debugging purposes and for the acquisition of metrics data such as CPU utilization or task context switch .Moreover, the SVF was instantiated many times over so that each developer can have his own copy. Allowing the developers to use a personal instance of the SVF was a critical aspect in the frame of the ESAIL project. This permitted to have pre-tested OBSW versions just ready for the AIT Flatsat campaigns. The ESAIL SVF facility modelled all relevant S/C interfaces visible to the OBSW. The simulation can be executed at real time or faster than real time, allowing the user to save time for debugging purposes. The SVF infrastructure and the models were enabled to allow the injection of transient or persistent errors used to verify the FDIR capabilities of the flight software. Simplified environmental models and satellite dynamics using predefined data were used to supply inputs to the sensors units. The schematic of the SVF-SW facility is shown in Fig 4.

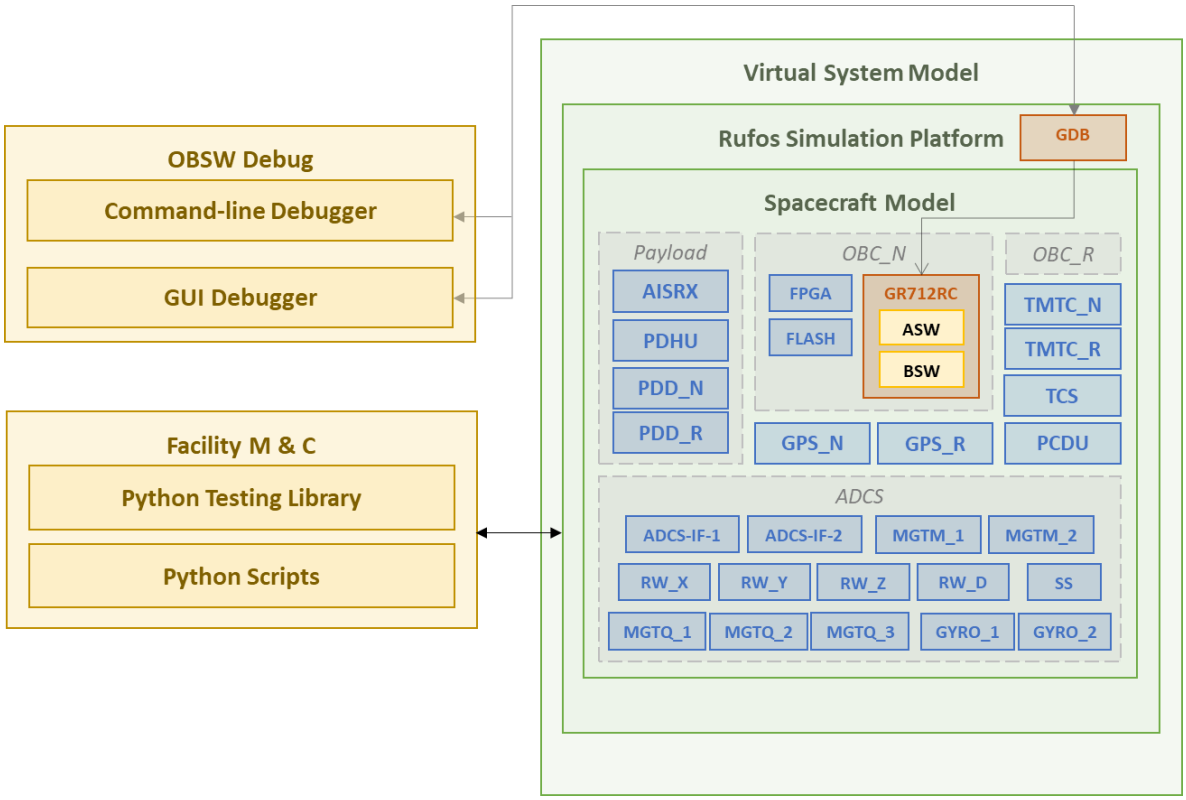

Fig 4 ESAIL SVF-SW facility

AIV Simulator (AIVS)

During the ESAIL qualification and acceptance, the ESAIL simulator was used as back-end integrated together with the Central Checkout System (CCS) used in the project (EGSE-SCOS). In such spacecraft AIV Facility configuration, the simulation of all the equipment, was implemented as part of this central simulator function however it also opened the door to integrate real hardware equipment. In ESAIL, the simulator models for this facility, were identical than the ones used in the SVF. TCL scripts were developed and executed using EGSE-SCOS TOPE environment, being the satellite commands and telemetries processed and generated by the AIV simulator.

Operational Simulator (TOMS)

The final facility implemented ESAIL is the Operations Simulator. In ESAIL this facility is a high fidelity model of the spacecraft that is obtained by model configuration. Representative simulation of the spacecraft telecommanding and housekeeping telemetry for the platform and the payload modules is provided. This facility includes the environmental and spacecraft dynamics models that have been converted from the Matlab/Simulink models of the ADCS FES Simulator to SMP2/C++ using MOSAIC 10 tool. The block diagram on the facility is given in Fig 5

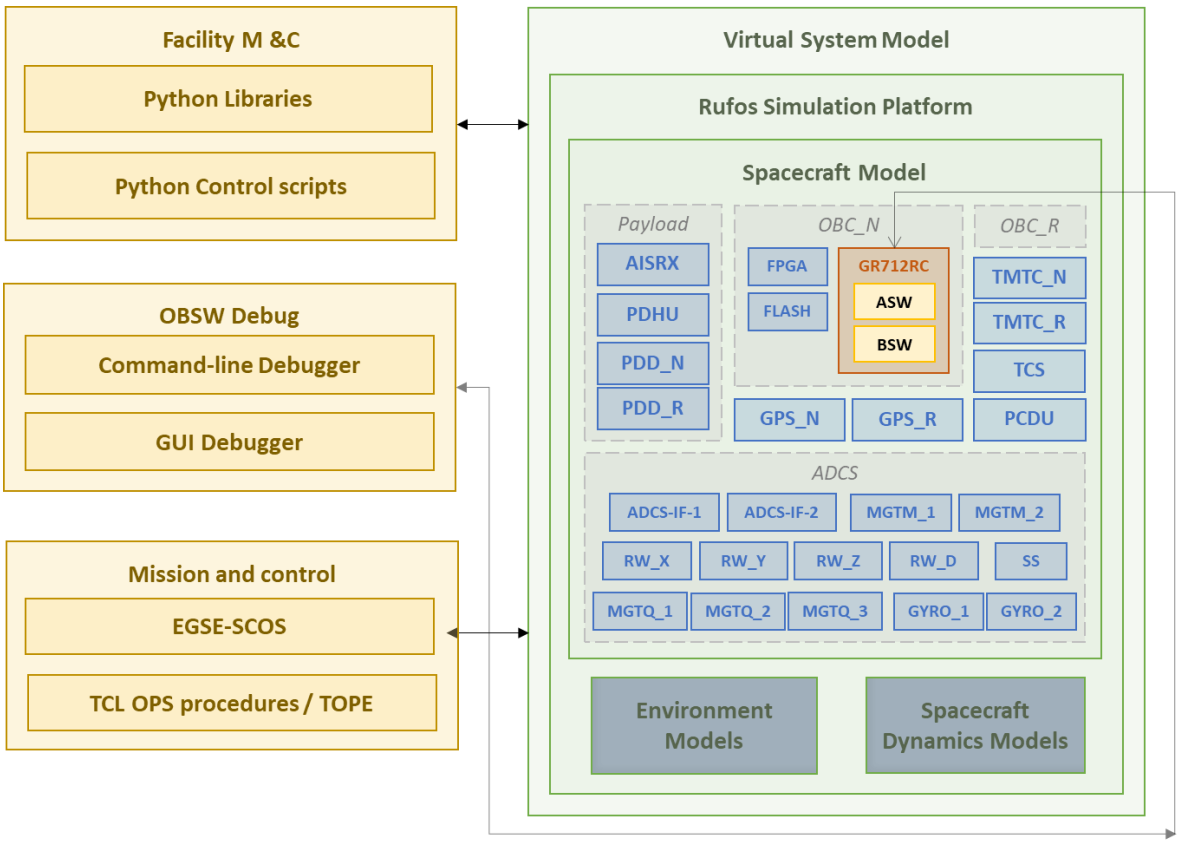

Fig 5. ESAIL TOMS facility

CONCLUSIONS AND FUTURE WORK

ESAIL is expected to be launched from Kourou during the second half of 2019. At the date of issue, the Operational Simulator is still on its last phase of development in close collaboration with ESA representatives. Further use-case extensions of the simulator are to be implemented, to facilitate ADCS close-loop simulation using the OBC Emulator and also the real OBC engineering model. The ESAIL simulator is expected to be used to support the operations team on maintenance duties during the satellite phase E.
The ESAIL multipurpose simulator constitutes the first complete SMP2 Simulator that has adopted the SMP2 standard at LuxSpace and has consolidated the use of a common simulation platform (Rufos) and strengthen the collaboration within the OHB group teams in charge of building simulation facilities.

REFERENCES

[1] ECSS-E-TM-10-21A “Space Enginiering – System modelling and simulation” 16 April 2010.
[2] Cobham Gaisler, *“A Dual-Core LEON3-FT SPARC V8 Processor Data Sheet”*, https://www.gaisler.com/doc/gr712rc-datasheet.pdf.
[3] A. Weihusen, P. Froehner, A. Trung, M. Gehre, D. Della Ratta, N. Lambl, D. Lammers, H. Lindberg, M.-S. Nizou, G. Robbers, I. Vukman “ OHB’s Software Base Simulator: Efficient Development of Software-based Simulators by Re-use of Generic Components“ Proceedings of SESP 2017
[4] P. Froehner, A. Gamarra, M. Gehre, A. Weihusen, F. Hoffmann, D. Della Ratta, “MTG SVF: An excellent opportunity for assessing the SMP2 compatibility“, Proceedings of SESP 2015, March 2015
[5] A. Trung, M. Gehre, P. Froehner, D. Della Ratta, N. Lambl, D. Lammers et al, “Developing a SMP2 compliant Hardware-In-the-Loop simulation framework”, Proceedings of SESP 2017
[6] „SMP 2.0 Metamodel“ EGOS-SIM-GEN-TN-0100, issue 1.2, 28.10.2005
[7] „SMP 2.0 Component Model“ EGOS-SIM-GEN-TN-0101, issue 1.2, 28.10.2005
[8] “SMP 2.0 C++ Mapping” EGOS-SIM-GEN-TN-0102, issue 1.2, 28.10.2005