OHB's Software Base Simulator: Efficient Development of Software-based Simulators by Re-use of Generic Components

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 System AG
Universitaetsallee 27-29
28359 Bremen, Germany
Email: andreas.weihusen@ohb.de, pamela.froehner@ohb.de

ABSTRACT

The simulation of satellite systems plays an increasing role in the support of several engineering and operational activities during the lifecycle of a satellite programme. In order to reduce the development effort, costs and risks for software-based simulators in satellite programmes, a high degree of re-use of simulator software items is desirable, between different project phases as well as between different projects.

This article presents OHB's approach to increase the efficiency of the software simulator development: A collection of qualified, generic software components is used as the base for the development of the mission specific simulator facilities, like SVF, AIVS or TOMS. This collection is denoted as the "Software Base Simulator". In particular, the Software Base Simulator consists of the simulation runtime environment Rufos and the so-called simulator platform models (i.e. models of the standard S/C system interfaces and buses as well as interface models for monitoring and control, calibration/configuration and debugging). It can be extended with models of standardised S/C equipment, e.g. the on-board computer. All simulator components are implemented according to ESA's SMP2 (Simulation Model Portability 2) standard to allow the re-use of the components in different simulator facilities.

A generic simulator software system specification (SSS) defines the requirements baseline (RB) of the Software Base Simulator. This SSS is traced down to the software requirement specifications (SRS) of the individual simulator components. The combination of these SRSs then defines the technical specification (TS) of the Software Base Simulator. The re-use of these specifications within specific S/C missions is explained in the article. Furthermore, the automated build and validation process as well as the basic configuration management approach for the Software Base Simulator development is described.

Finally, the article shows how the Software Base Simulator is extended to a full-featured simulator facility for a specific satellite mission.

INTRODUCTION

Software-based simulations of a satellite system are used throughout the complete lifecycle of the satellite program for a variety of applications: *Software Validation Facilities* (SVF) support the development and validation of the mission-specific satellite control software (SCSW); *Assembly, Integration and Verification Simulators* (AIVS) provide emulations of non-available hardware in hardware-in-the-loop configurations; *Training, Operations and Maintenance Simulators* (TOMS) are applied in Spacecraft Control Centers for the named purposes.

OHB gained experience in the simulator development over several years by developing software-based simulators for SAR-Lupe, EnMAP, SmallGEO, EDRS-C and MTG (for Galileo a hardware-based approach was applied). Initially, the specific simulators were developed independently and a re-use of simulator components happened only partially, e.g. between SAR-Lupe and EnMAP or between SmallGEO and EDRS-C. The MTG SVF was the first simulator facility that was implemented according to the SMP2 standard to allow a re-use of simulation models in following simulator projects.

The implementation of a complete software-based simulator facility from scratch entails a considerable amount of development time, cost and risk. The re-use of qualified and approved software components within the different simulator facilities of one project and also between different satellite projects helps to increase the efficiency of simulator development and to minimize the development risks and overall costs of a project. The main candidates for re-use are mission-independent, generic software items, which are required for each simulator facility, like the simulation runtime environment, simulations of the satellite electrical interfaces and also simulations of HW equipment that is procured for several missions. OHB denotes this set of qualified generic software components as its "Software Base Simulator" and uses it as the initial setup for the development of current and future software-based satellite simulators. Details on this Software Base Simulator are given in the following sections. The required steps to extend the Software Base Simulator to a full-featured simulator facility for a specific satellite mission are described in the corresponding sections as well.

SIMULATOR COMPONENTS

The Software Base Simulator denotes a collection of qualified, project-independent software components that is used as the base for the development of the mission-specific simulator facilities at OHB, which are SVF, AIVS and TOMS. The software components can be categorised into the simulation runtime environment **Rufos**, the **Platform Models** and the **Common Models**. In order to ensure the portability and re-use of these components, each of them is implemented according to ESA's SMP2 (Simulation Model Portability 2) standard, as currently specified in [1], [2] and [3]. This standard was introduced at OHB for the development of the MTG SVF and it has proven to be an appropriate basis for the future simulator development, as described in more detail in [4]. The functionalities of the particular components of the Software Base Simualtor are described hereinafter.

Rufos (Runtime for Simulations) is OHB's SMP2 compliant simulation runtime environment. It hosts the simulation models of the simulator facility and provides all necessary services to configure and control these models during a running simulation. It implements the different simulator states and the mandatory simulation services like Logger, Time Keeper, Scheduler and Event Manager and also the optional Resolver service as specified in [2]. It also provides interfaces to control the simulation via a Man Machine Interface (MMI) for interactive simulations or via scripts for the execution of automatic procedures. Rufos is defined as a stand-alone software item with a related documentation set and configuration item (CI) number.

Rufos was introduced in [4] for the SESP workshop 2015. It will be used as the baseline for all upcoming simulator projects at OHB, unless required differently by the customer. Its software design allows the compilation for currently two target operating systems, RHEL/CentOS 7 (for SVF and TOMS) and the real time operating system QNX (for AIVS, see [5]).

The **Platform Models** denote a set of SMP2 compliant models, which implement generic functionalities that recur within the different simulator facilities. This set of models is defined as one stand-alone software item with a related documentation set and configuration item (CI) number. It includes the following software elements:

- The **Generic model** implements the base class, from which all further simulation and interface models are derived.
- The so-called **Line Models** implement simulation models of the standard electrical interfaces of spacecraft systems, such as MIL-STD-1553 bus, SpaceWire, CAN bus and discrete signal lines.
- The **Calibration Service** loads raw-to-engineering and engineering-to-raw conversion curves from MIB (mission information base) files, which are usually exported from the project's satellite reference database (SRDB). These calibrations are then provided to the simulation models via an SMP2 service.
- 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 it exchanges telecommands (TCs) and telemetries (TMs) using a determined encapsulation protocol. For SVF and AIVS a central checkout system (CCS) may be used for monitoring and controlling the simulation or for the execution of automatic test sequences, while the TOMS is connected to the mission control system of the space control centre. Besides the connection to the M&C facility, the simulation can be also controlled using the user and scripting interfaces provided by Rufos.

- A **GNU Debugger (GDB) Interface Model** provides the connection to an external tool for debugging the satellite control software (SCSW) in the SVF. Although this model is exclusively used in the SVF, it is nevertheless generic and re-usable in the individual mission-specific SVFs.

The **Common Models** denote SMP2 compliant simulation models of standardized spacecraft equipment that is intended for usage in various missions. Due to this equipment "re-use" it is obvious to make the corresponding simulation models available as part of the Software Base Simulator. Unlike the Platform models, each common model is treated as a stand-alone software item, provided with its own documentation set and CI number.
The initial common model is a simulation model of the **Satellite Management Unit** (SMU), also denoted as onboard computer. This model simulates the SMU with high fidelity, such that the real SCSW images can be run with cycle accurate behaviour. In order to achieve this, the SMU simulation model includes an emulation of the target processor instruction set as well as simulation models of the SMU internal hardware that are relevant to the execution of the SCSW. In the SVF, the SMU model is used to host unmodified versions of the SCSW for development and validation purposes as well as for testing the FDIR behaviour of the SCSW. In the TOMS, the SMU model hosts the SCSW that is also running on the satellite itself.
Further common models are currently a simulation model of a micro **Remote Terminal Unit** (µRTU) and a simulation model of a **Global Navigation Satellite System Receiver** (GNSSR).

The particular Software Base Simulator components were validated in the scope of the simulator projects, in which they were applied first. The corresponding software validation specifications and reports are part of the accompanying documentation set that is put under configuration control, as described later on.

An example for the generic architecture of a software-based satellite simulator is shown in Fig. 1, including Rufos (light blue), the platform models (dark blue) and the common models (light green, represented by the SMU simulation model). The base components are complemented by the project-specific simulation models (light yellow) and supplied/external components (orange).

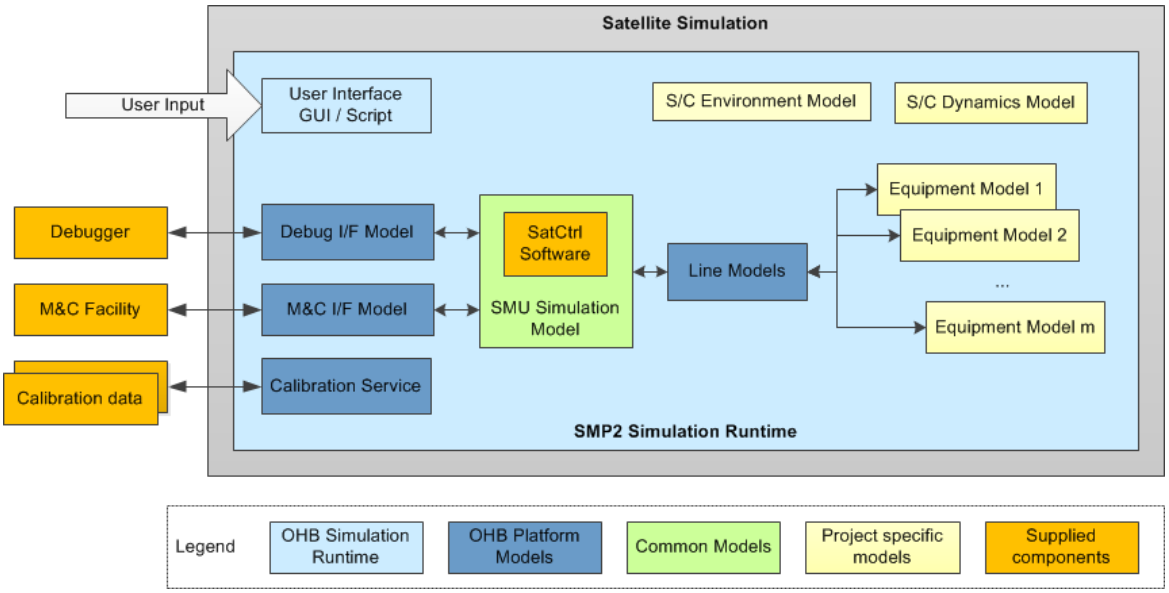

Fig. 1 Generic SW Simulator Architecture

REQUIREMENTS

A generic simulator software system specification (SSS) defines the requirements baseline (RB) of the Software Base Simulator. In this SSS, the generic simulator functionalities are specified. A specific attribute assigns the applicability of the requirements for SVF, AIVS and/or TOMS. Some examples: Requirements for SCSW debugging may only be

applicable for the SVF; the real-time requirements of an AIVS are usually different to the ones for SVF and TOMS, while the requirements of the MIL-STD-1553 bus simulation model are applicable for each of these facilities.

The generic SSS requirements are re-used in the SSS for a project-specific simulator facility in the following way: following to the analysis and selection of the generic requirements that shall be re-used, these requirements are included (i.e. copied) into the project-specific simulator facility SSS with their original requirement identifier. These requirements are then complemented by the project-specific simulation requirements, which have a project-specific requirement identifier. The different requirement identifiers allow an easy identification of the requirements that are re-used and thus already validated.

The generic Simulator SSS is traced down to the software requirement specifications (SRS) for Rufos, Platform Models and the individual Common Model SRSs. The combination of these SRSs define the technical specification (TS) of the Software Base Simulator. According to the re-use of the generic RB requirements in the project-specific SSS these requirements are already satisfied by the corresponding TS requirements for Rufos, Platform Models and Common Models, thus the traces from TS requirements to RB requirements can be re-used in the project-specific traceability matrices as well. The TS of the project-specific simulator facility will be completed by a corresponding specific simulator SRS, which includes the software requirements for the project-specific simulation models. The relations between the generic and project-specific specifications are shown in Fig. 2.

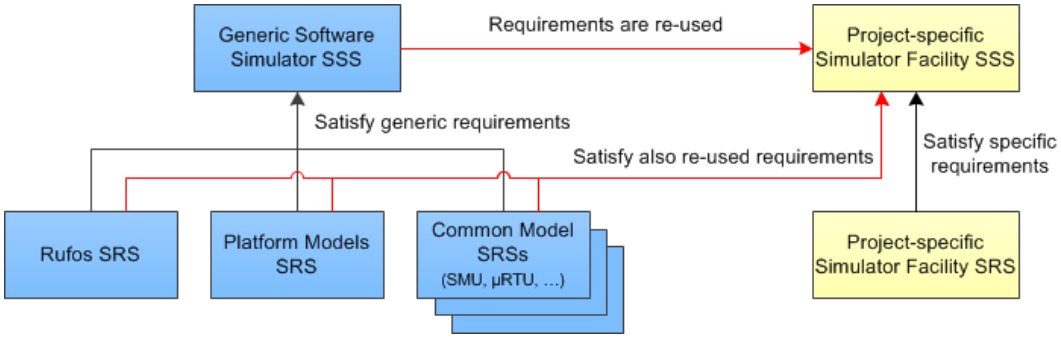

Fig. 2 Specification Tree

CONFIGURATION MANAGEMENT

OHB uses the version control system *Git* for the configuration control of its simulator software. This includes both the project-specific software items and the items of the Software Base Simulator. Projects and Base Simulator components are grouped on the same level in the repository structure. For Rufos, the related libraries, binary files, configuration files, installation and administration packages and graphical components are stored, from which the runtime can be built. Each simulation model is stored with its SMP2-specific catalogue, assembly, schedule and configuration files, its source code, build environment and build artefacts and also with its corresponding test scripts. This applies to each particular model of the Common Models component as well as to the Platform Models component as a whole. The project-specific folders include the configuration data for the respective simulator facilities and the specific equipment simulation models.

Each software item of the Software Base Simulator has a related set of documents to reduce the documentation effort in the re-use case. This set usually includes the SRS, the interface control document (ICD) and the software design document (SDD) of the respective software item, together with the related test documentation (including the validation specification and the validation and verification report), the user manual and the software configuration file.

BUILD PROCESS

The project-specific simulators are built using the continuous integration system Jenkins. Each simulator facility is composed of two rpm packages (rpm is a standard package format on *Red Hat*-based Linux systems), one for the simulation runtime environment Rufos and one for the particular software items that implement the specified behaviour

of the simulator. Both packages are generated independently from the simulator repository. The Rufos rpm is built from the Rufos directory, while the simulator rpm incorporates all required models from “Platform”, “Common Models” and the project-specific models as well as the related assembly and configuration files and test scripts. The contents of each individual simulator rpm are defined via corresponding configuration files for Jenkins.

After the creation of the two rpm files, Jenkins installs the target simulator facility from these packages and executes the test scripts that belong to the incorporated software items. At the end of the automated test execution, the integration system generates the corresponding parts of the validation reports with respect to the TS and the RB from the test results. If all tests are successfully executed (and if the remaining manual validation and verification activities are completed as well), the simulator facility, consisting of both rpm packages, can be released by the project leader according to the project schedule and upon customer request.

CONCLUSION

The Software Base Simulator components are already used within the current simulator development activities at OHB:

- Rufos, Platform models and the SMU simulation model provide the base of the initial SVF version for SARah, supplemented with a small set of equipment simulation models. This initial SVF allowed starting the SCSW development already at an early stage of the SARah project. It will be completed incrementally with the remaining models to full functionality in correspondence with the SCSW schedule. In addition to the SVF development, Rufos and Platform models were compiled on the real-time operating system QNX to build the initial version of an AIVS for SARah, as described in detail in [5].
- Similarly to the SARah SVF, the initial version of the Electra SVF is created on the basis of the Software Base Simulator and provided to the SCSW team to start the development.
- Moreover, Rufos and Platform models were provided to LuxSpace and allowed building up an SVF for the E-Sail project in a very short time. The provision of the Software Base Simulator for projects of other companies of the OHB group is currently being analysed.

It has been shown by these particular projects that the usage of the Software Base Simulator accelerated the development time of the required simulator facilities remarkably. Furthermore, the re-use of these components reduced the effort for validation, documentation and configuration management as well. Thus, it can be stated that the described approach indeed leads to an increased efficiency of the simulator development.

REFERENCES

[1] „SMP 2.0 Metamodel“ *EGOS-SIM-GEN-TN-0100*, issue 1.2, 28.10.2005

[2] „SMP 2.0 Component Model“ *EGOS-SIM-GEN-TN-0101*, issue 1.2, 28.10.2005

[3] “SMP 2.0 C++ Mapping” *EGOS-SIM-GEN-TN-0102*, issue 1.2, 28.10.2005

[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”, in press for *Proceedings of SESP 2017*