MTG SVF: AN EXCELLENT OPPORTUNITY FOR ASSESSING THE SMP 2.0 COMPATIBILITY.

Pamela Froehner, Amaya Gamarra, Matthias Gehre, Andreas Weihusen, Finn Hoffmann, Domenico Della Ratta

OHB System, SW Department, Simulation division, Karl-Ferdinand-Braun-Str.8
28359 Bremen, Germany.

Contact: Tel : +49-(0)421-2020 721, e-mail : pamela.froehner@ohb.de

ABSTRACT

SMP 2.0 (Simulation modelling platform) is ESA's standard for reusable simulation models in space related simulators. The paper refers to the currently applicable SMP 2.0 which is approved and defined in the set of EGOS-SIM technical notes[1].

This article presents OHB's experience working with SMP 2.0: within the frame of the development of the MTG SW validation facilities, which require SMP 2.0 compliant simulation models, the OHB simulation team performs an assessment of the features offered by the different SMP 2.0-compliant simulation frameworks. OHB establishes a set of objective project choice criteria leading to the selection of Basiles (Simulation environment owned by CNES and developed by Spacebel) as simulation environment for the MTG Software Validation Facility, SVF. Before this choice has been made, the model development is partially subcontracted and only the compliance to SMP 2.0 can be requested to the suppliers. The SMU Simulation model is then developed and first tested on SIMSAT.

During the design of some simulation models OHB discovers that the full compatibility ensured by SMP 2.0 does not allow a simple plug and play operation of simulation models among the different SMP 2.0 compliant environments. The paper exhibits the encountered difficulties, the adopted solutions and intends to list the design decisions to be taken to ease the reusability of models among the different SMP 2.0 compliant environments.

On a second step the article presents the encountered limitations on Basiles and SIMSAT that led OHB to the development of Rufos (Runtime FOr Simulation), an OHB own SMP 2.0 environment. Rufos is developed to support the SMP 2.0 compliant inhouse development and debugging of models for the MTG SVF. The whole formal validation process is performed on the target simulation environment (Basiles). Rufos only implements the services included in the currently applicable SMP 2.0 specification and that greatly simplifies the configuration and use of the simulator. In addition, as Rufos does not need to keep backwards compatibility with any previous OHB simulation framework, its design can be exclusively focused in obtaining the maximum yield of the computer resources. Considering the increasing demand of fidelity requested to the simulation models and the rise of programs with high demanding processor emulators (e.g. Sarah), the optimization of computer resources becomes a matter of urgency.

Last but not least, the paper presents what is the way forward that OHB intends to follow for simulator facilities use on future programs.

[1] EGOS-SIM-GEN-TN-0099 SMP 2.0 Handbook, version 1.2, 28.10.2005,
EGOS-SIM-GEN-TN-0100 SMP 2.0 Metamodel, version 1.2, 28.10.2005
EGOS-SIM-GEN-TN-0101 SMP 2.0 Component Model, version 1.2, 28.10.2005
EGOS-SIM-GEN-TN-0102 SMP 2.0 C++ Mapping, version 1.2, 28.10.2005
EGOS-SIM-GEN-TN-1001 SMP 2.0 C++ Model Development Kit, version 1.2, 28.10.2005

INTRODUCTION

The Simulation modelling platform specification has been first presented under the name of SMP 2.0 and detailed in the set of EGOS-SIM-GEN technical notes. Despite the SMP simulation standard (ECSS-E-TM-40) is still under approval, this first specification has propelled the stakeholders of the European Space industry to use the SMP 2.0 technology as a mean to ensure the reuse of the simulation models among programs. With the aim of advancing in the standardization, ESA has invested a considerable effort in the development of SIMSAT, the first simulation infrastructure fully SMP 2.0 compliant and the distinct satellite prime contractors have adapted their own simulation environments to this specification.

Nowadays in the space industry, we can currently count almost so many different simulation environments as satellite prime contractors. Replacing the simulation environment has, at industry side, a big impact on the development and test activities. On the one hand it requires an adaptation of the in house existing simulation models and their test procedures, on the other, validation teams have to be trained to the new testing features and language. For these reasons, most of the prime contractors have just adapted their non-SMP 2.0 born environments to SMP 2.0 instead of just using SIMSAT.

On the other side, it’s a proven fact that the current SMP 2.0 specification is not mature enough leading to different interpretations and thus the simulation environments are not identical. In the case of SIMSAT, ESA has been forced to take some design decisions which, unfortunately, have not been traced in the handbook and thus not shared by the primes in the adaptation of their environments. In addition, at industry side, when technical incompatibilities have emerged among the existing environments and SMP 2.0, the non-regression of the existing simulation features has, in many cases, weighted more in the design of the solution.

OHB bursts onto this SMP 2.0 simulation stage with the development of the Satellite Validation Facility for MTG program. Without any expertise on this technology and due to the lack of SMP2.0 simulation environment of its own, OHB is then faced, first to choose an existing SMP 2.0 compliant simulation environment and later with the integration of different SMP 2.0 simulation models which have been developed for and tested on different environments.

This paper relates the story of the integration, on two different SMP 2.0-compliant simulation environments, of SMP 2.0-compliant models which were initially developed targeting a, still distinct, third SMP 2.0-compliant framework.

THE TARGET SMP 2.0 ENVIRONMENT

As mentioned, OHB does not own any SMP 2.0 simulation environment easily reusable among spacecraft programs and therefore, the selection of this tool within the frame of MTG program can be handled with complete impartiality. Upon assessment of the existing tools it is clear that the features provided by the different simulation environments are very close. The main differences lie on the scripting language that may be used for developing the test procedures and on the features of the Man Machine Interface (MMI). The latter may be important for an operational simulator but are not so significant when we talk about a facility for testing the spacecraft software on which the tests are mainly automated and therefore often run in batch mode.

As a reminder, the SMP 2.0 specification identifies the following services to be implemented by a compliant environment:

- A Simulator component to manage the simulation states and the access to the other services;
- A Logger component to centralize the log/traces generated by the simulation models, services and test procedures;
- A Time Keeper component to keep track of the simulation time;
- A Scheduler component to manage the schedule events;
- A Resolver service which plays the role of a name server and allow referencing an instance by its path;
- An Event manager component to handle the notifications of framework events;

These components are the “greatest common divisor” of all simulation environment that can then be, as mentioned, powered with more or less graphical features. In the case of MTG SVF project, Basiles has been selected and the criteria weighting the most on this decision are:

- Basiles is powered with a TCL interpreter which simplifies the reuse of test procedures between the SVF and the EGSE (powered with a uTOPE user interface).
- Basiles licensing is a much simpler process than SIMSAT which may need more than one year.

THE SMP 2.0 SMU SIMULATION MODEL

The so called Software Management Unit (SMU) is the on-board computer of the MTG platform. The procurement of the SMU simulation model was advanced in the time regarding the procurement of the simulation environment and therefore, no requirements targeting the environment in which the SMU model had to be integrated could be included in the SMU model contract. Only a requirement imposing compliance to SMP 2.0, “potentially” ensured a simple integration of the SMU model in the MTG simulator.

Terma, the contractor responsible for the SMU model design and development, had a large background in SIMSAT and thus this was the target environment for the validation of the model against the technical specification.

THE INTEGRATION OF THE SMU SIMULATION MODEL: SWITCHING FROM SIMSAT TO BASILES

Already during the conception phase of the SMU simulation model, its design was constrained by its integration in Basiles and the following implementation arrangements were made to the model with the aim of mitigating the integration risks:

- Prior to starting the development of the SMU model, Terma developers received the set of header files with the SMP 2.0 data type declarations from Basiles which are generated according to the contents of the EGOS-SIM-GEN technical notes. The SMP 2.0 data types used by SIMSAT seem to include minor evolutions with regard to what is specified, as a matter of example, the type Smp::Mdk::Array has a SIZE member in SIMSAT, but neither in Basiles nor in the definitions provided in EGOS-GEN-TN.
- The SMP 2.0 simulator events are defined without duration (Discrete Event Simulation, DES), what compels each emulated instruction to be scheduled separately in spite of impacting the simulation performances. It is empirically proven that the best simulation performances are reached when scheduling batches of about 250.000 emulator instructions, however this approach hinders executing in the right order any sporadic event being raised by a non-cyclic model, for instance from the I/O system. To solve this issue, the SMU model implements an IRunnable interface allowing the simulation environment (Basiles) to pause the execution of the emulator instructions when required and to resume it after the execution of the sporadic events. This interface is handled by Basiles but not by SIMSAT and therefore, it could not be validated by the SMU contractor.

For its integration in Basiles, a preliminary version of the SMU model was used. Despite of the adaptations above mentioned, the first steps on this process were slow and a bit daunting. The first blocking issue came from the assembly files format. The definition of the model instances, its links and the default values of the SMU model fields were implemented in C++, while in the remaining simulation models, this definition was done with SMDL assembly files. Both methods are SMP 2.0 compliant but they are incompatible to each other, the instances defined in C++ cannot be referred from a SMDL assembly file for establishing, for instance, a link and vice versa. As in the same simulator it is not possible to use both implementations, OHB was constrained to develop a tool for creating SMDL files from the SMU assembly definitions.

Another setback contributed to increase the effort and the time required to integrate the SMU model, the scripts developed by Terma and used to run the model on their development environment were useless on Basiles. SIMSAT embeds a Javascript interpreter while Basiles only a TCL one. The whole set of Terma scripts had to be translated.

Once these problems were overtaken, the execution of the SMU model on Basiles was possible, the test campaign first run on SIMSAT was rerun on Basiles and the results could be reproduced. It's important to highlight that these tests are designed by RUAG, the manufacturer of the SMU HW and that prior to having them run on the SVF, they were run and tuned on the HW. The aim of this testing approach was thus to check the fidelity of the simulator.

BACKWARDS COMPATIBILITY: SWITCHING FROM BASILES TO SIMSAT

During some moments of the SVF test campaign and with the aim of clearing the simulation environment as the cause of a software problem, OHB tried unsuccessfully to run the SVF on SIMSAT. The SVF configuration, at that stage of the project, already included the AOCS and EPS simulation models and for each subsystem a SMDL assembly had been generated. Between the EPS and SMU SMDL assemblies there were some circular references, id est the EPS assembly file referenced some instances defined in the SMU simulation model assembly and vice-versa. SIMSAT applies the configuration from each assembly file just after loading it, without waiting the load of the whole set of assemblies. For this reason it can only create links among instances that have been previously defined and does not support circular references. Handling a single assembly file for the whole SVF would be a possible workaround but this approach is not optimal from software configuration management point of view, since the file may be often submitted to changes and the configuration of the subsystems could then hardly be frozen. On the other hand, one major goal of SMP 2.0 is to make models self-contained and independent of each other for easy reuse and having a single assembly for the whole simulator also complicates reusability.

In addition, as mentioned above, SIMSAT does not implement the IRunnable interfaces and therefore the behavior of MTG SVF simulator upon integration of the AOCS models differs from the behavior observed in Basiles.

ASSESSMENT OF BASILES AS MTG SVF SIMULATION ENVIRONMENT

The MTG SVF is using Basiles as a simulation environment since more than one year and the main outcome is that this tool is highly reliable to run SMP 2.0 simulators given the fact that it has been proven to be fully compliant with the SMP 2.0 specification. The first SVF version has been successfully validated with no defects detected on the simulation environment.

However, Basiles Man Machine Interface (MMI) is not optimal for the development and integration phases of the S/C software and of the simulation models. Its usability can still be improved on the following topics:

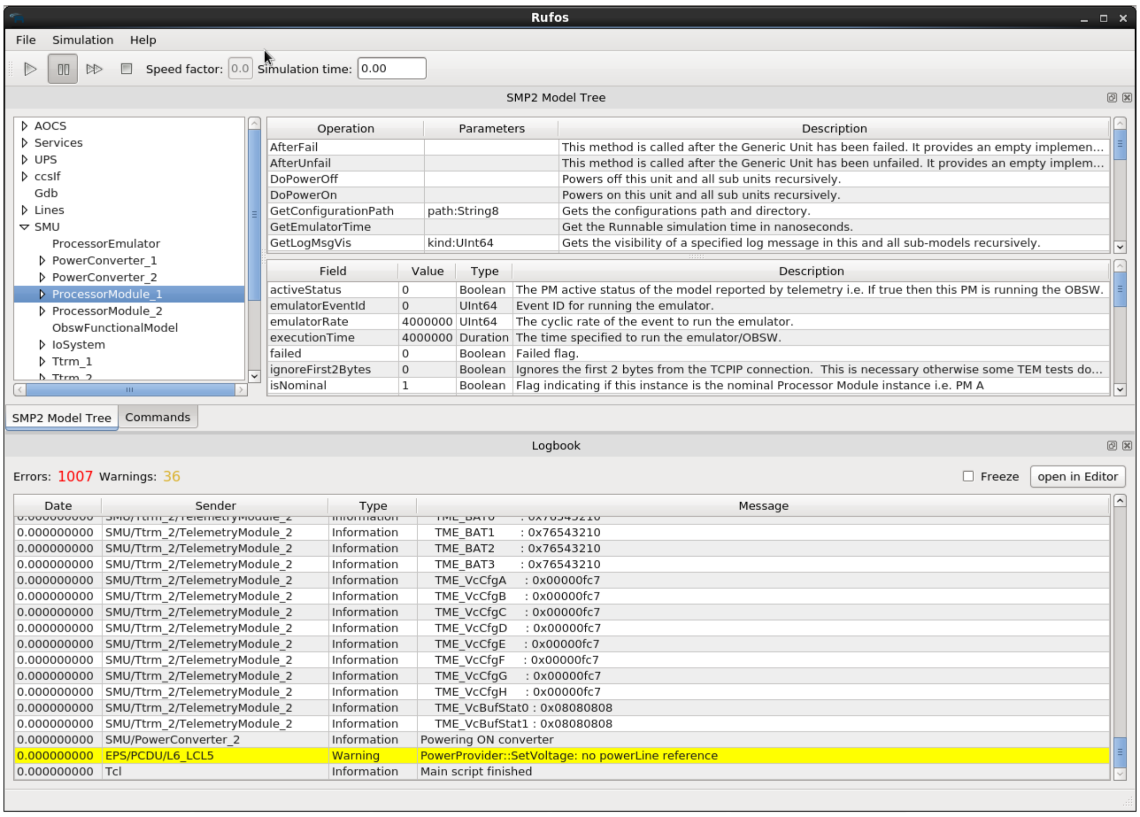
- Basiles requires around 15 seconds for loading the simulation and starting the TCL test script. This delay has been measured on a SVF including SMU, EPS and AOCS simulation models. To this delay the developer needs to add the time required by the TCL test script to be executed.
- In case of an error in a TCL test script, the step by step execution and the backtraces only reach the first scripting level. Stepping into procedures or providing a full backtrace is not possible This really complicates debugging of test libraries and therefore increases the learning curve of the validation teams. This limitation leads the developer to implement a debugging system based on logs, fact that combined to the delay required to load the simulator, may result in long and heavy campaigns of models development.
- Basiles includes an interface to the GDB debugger that allows debugging of the simulation models with the command-line GDB client. Connecting a graphical debugger is not straight forward due to the multi-process architecture of this environment.
- Basiles MMI is developed in TK, a graphical user interface toolkit for TCL. Programs developed with scripting languages often present poorer performances than programs developed with lower level languages. In this case, the user can feel a low responsiveness upon execution of some MMI features. It is the case of the so called introspection view which needs 3 to 4 seconds to present all the model instances intervening in the simulation.

- Basiles has been developed and qualified according to its classification of software with criticality D and therefore the robustness to malformed input has not been intensively tested. In some occasions, this tool does not provide the user with messages pointing to the origin of the failure.

- Basiles was not born as a SMP 2.0 simulation environment, the SMP 2.0 compliance was an add-in to the tool. Actually, Basiles implements some wrappers to the SMP 2.0 models to finally execute them on its native architecture. For this reason, Basiles needs to build the SMP 2.0 models with its own custom build system before running them. This fact increases the learning curve for the simulation integration but also extends the time required to validate and debug the models.

RUFOS

During the first steps of the SVF integration phase the Basiles MMI drawbacks had a strong weight on OHB integration and validation efforts. Basiles is owned by CNES and has been conceived for its needs, for this reason the adaptation of Basiles to OHB needs does not appear as a sensible solution. A close look at EGOS-SIM-GEN technical notes, led OHB to the idea of developing a light SMP 2.0 environment to ease the debugging tasks.



Operation	Parameters	Description
AfterFail		This method is called after the Generic Unit has been failed. It provides an empty implemen...
AfterUnfail		This method is called after the Generic Unit has been unfailed. It provides an empty implem...
DoPowerOff		Powers off this unit and all sub units recursively.
DoPowerOn		Powers on this unit and all sub units recursively.
GetConfigurationPath	path:String8	Gets the configurations path and directory.
GetEmulatorTime		Get the Runnable simulation time in nanoseconds.
GetLogMsgVis	kind:UInt64	Gets the visibility of a specified log message in this and all sub-models recursively.

Field	Value	Type	Description
activeStatus	0	Boolean	The PM active status of the model reported by telemetry i.e. If true then this PM is running the OBSW.
emulatorEventId	0	UInt64	Event ID for running the emulator.
emulatorRate	4000000	UInt64	The cyclic rate of the event to run the emulator.
executionTime	4000000	Duration	The time specified to run the emulator/OBSW.
failed	0	Boolean	Failed flag.
ignoreFirst2Bytes	0	Boolean	Ignores the first 2 bytes from the TCPIP connection. This is necessary otherwise some TEM tests do...
isNominal	1	Boolean	Flag indicating if this instance is the nominal Processor Module instance i.e. PM A

Errors: 1007 Warnings: 36

```
Date         Sender                        Type          Message
0.000000000  SMU/Ttrm_2/TelemetryModule_2  Information    TME_BAT1      : 0x76543210
0.000000000  SMU/Ttrm_2/TelemetryModule_2  Information    TME_BAT2      : 0x76543210
0.000000000  SMU/Ttrm_2/TelemetryModule_2  Information    TME_BAT3      : 0x76543210
0.000000000  SMU/Ttrm_2/TelemetryModule_2  Information    TME_VcCfgA     : 0x00000fc7
0.000000000  SMU/Ttrm_2/TelemetryModule_2  Information    TME_VcCfgB     : 0x00000fc7
0.000000000  SMU/Ttrm_2/TelemetryModule_2  Information    TME_VcCfgC     : 0x00000fc7
0.000000000  SMU/Ttrm_2/TelemetryModule_2  Information    TME_VcCfgD     : 0x00000fc7
0.000000000  SMU/Ttrm_2/TelemetryModule_2  Information    TME_VcCfgE     : 0x00000fc7
0.000000000  SMU/Ttrm_2/TelemetryModule_2  Information    TME_VcCfgF     : 0x00000fc7
0.000000000  SMU/Ttrm_2/TelemetryModule_2  Information    TME_VcCfgG     : 0x00000fc7
0.000000000  SMU/Ttrm_2/TelemetryModule_2  Information    TME_VcCfgH     : 0x00000fc7
0.000000000  SMU/Ttrm_2/TelemetryModule_2  Information    TME_VcBufStat0 : 0x08080808
0.000000000  SMU/Ttrm_2/TelemetryModule_2  Information    TME_VcBufStat1 : 0x08080808
0.000000000  SMU/PowerConverter_2          Information   Powering ON converter
0.000000000  EPS/PCDU/L6_LCL5              Warning       PowerProvider::SetVoltage: no powerLine reference
0.000000000  Tcl                           Information   Main script finished
```

Fig. 1. Rufos Man Machine Interface

OHB’s SMP 2.0 environment is now a fact and receives the name of Rufos (RUntime FOr Simulation). It is built on the basis of a light architecture on which only the SMP 2.0 services are implemented. The IRunnable interface to control the emulator execution is also implemented, which allows the safe execution of MTG SVF.

Of course this simulation environment is model-development oriented and implements a lean but fast MMI (see Fig. 1). Both models as well as test scripts can be easily debugged with a graphical debugger (e.g. Eclipse). With the aim of ensuring a full compatibility with MTG, it integrates a TCL interpreter. Additionally, a Python interpreter has also been integrated for simplifying the daily tasks of the S/C software developers, which in most cases already knew this language.

As Rufos is still not qualified, currently its use is strictly internal to OHB, as a matter of example, the MTG validation campaigns of the SVF and of the S/C software are always executed in Basiles. However, Rufos qualification is planned and the OHB’s intention is to use it as baseline for all simulators projects.

CONCLUSION

SMP 2.0 specification is a good start but still cannot ensure plug and play portability of models among simulation environments. Considering this feature is one of the main goals of this specification, it should ensure that when different implementation options are offered, they are compatible to each other.

In what concerns the simulation environment, if we gather 1000 engineers and ask them to list the features for the perfect MMI we would be able of designing, at least, 900 MMIs. The MMIs are designed on the basis of subjective needs and the efforts that an organization may need for using an interface not adapted to its needs may be higher than the efforts of developing a new one from scratch.

In addition, the time and effort nowadays required to get the licenses of Simsat is a risk that has to be handled individually in every project, fact that also has a discouraging effect when the choice of simulation technologies is done at company level.

For all these reasons, as the objective of unifying a particular simulation environment in the industry seems hard to reach, it gives a cause for reflection on whether it is not better to concentrate the efforts in achieving an interface specification for simulation without gaps in respect of the portability.