A Methodology for Effective Reuse of Design Simulators in Operational Contexts: Lessons Learned in European Space Programmes

11th International Workshop on Simulation & EGSE facilities for Space Programmes SESP 2010

28-30 September at ESTEC, Noordwijk, the Netherlands

Cristiano Leorato[(1)], Peter van der Plas[(2)]

[(1)] Rhea System c/o ESA/ESTEC,
Keplerlaan, 1 - 2201 AZ Noordwijk, the Netherlands,
Email:cristiano.leorato@esa.int

[(2)]ESA/ESTEC
Keplerlaan, 1 - 2201 AZ Noordwijk, the Netherlands,
Email: Peter.van.der.Plas@esa.int

INTRODUCTION

The development of operational simulators is commonly based on the reuse of components across different missions (e.g. standard run-time simulation infrastructures, generic models). In the last few years, specific ESA missions have implemented a complementary reuse strategy, focusing on the reuse of simulators across the mission life-cycle:

- In the Automated Transfer Vehicle (ATV) mission, the design simulators of major subsystems have been reused as part of the real-time ATV Test Facilities. ATV Test Facilities include, among others, the Functional Simulation Facility (FSF), the Software Validation Facility (SVF), and the ATV Ground Control Simulator (AGCS). The ATV Test Facilities have been developed by ASTRIUM ST. The first ATV, Jules Verne, was launched in March 2008 and four more ATVs will be launched until 2015.
- In the Lisa Pathfinder (LPF) mission, the DFACS (Drag Free Attitude and Control System) design simulator has been reused and is now one of the core components of the LISA Pathfinder simulator for the Science Technology & Operations Centre (STOC), located at ESA/ESAC. LISA Pathfinder is scheduled for launch in 2012. The DFACS Simulator was developed by ASTRIUM Gmbh. It was originally conceived for the analysis of the Drag Free and Attitude Control System performance, and served as a prototype for the on-board control algorithms.

In the near future, the simulator for the PROBA3 mission is foreseen to support a wide range of phases in the project life-cycle. It will use the Formation Flying Test Bed (FFTB), which provides the means to develop simulators addressing the specific needs of formation flying missions. FFTB also supports the reuse of Matlab/Simulink models, usually developed as part of the initial system design, towards later phases, including software validation and operations.

This paper reports on recent ESA experiences in the reuse of simulators across the project life-cycle. It especially focuses on the reuse of design simulators in an operational context. Guidelines are identified, allowing understanding, at early phases, the technical implications of reuse. The applied methodology can also give indications to the management, helping assess the actual feasibility and cost of the reuse of the existing simulators.

DESIGN SIMULATORS AND OPERATIONAL SIMULATORS

Design simulators are developed at the beginning of the project life-cycle, in order to support the design of the space mission, e.g. to evaluate performances and the budget for critical resources (mass, power, etc.). They can be developed at different levels, and their actual scope is very diverse, as it ranges from:

- Simulation of the algorithms to be implemented in the on-board software.
- Simulation of one or more hardware subsystems.
- End-to-end simulation, including both the ground segment and the flight segment.

Operational simulators are developed in phases C/D, in order to:

- Support the preparation and validation of the operational procedures.
- Support the training of the operators at the Control Centre.

Except for end-to-end simulators (too large in scope), design simulators are a potential candidate to be part of the operational infrastructure. A number of challenges immediately arise:

- Design simulators and operational simulators are often developed and maintained independently.
- Technologies are very different. Design simulators often use Model Based technology (typically Matlab/Simulink), whereas operational simulators are built upon component oriented technology, the Simulation Model Portability (SMP2) standard, and Reference Architecture concepts (refer to [4] and [5]).

THE LISA PATHFINDER STOC SIMULATOR

During the operational phase of the LPF mission, the STOC will be in charge of the operations of the LPF experiments. For the STOC to be able to perform its planning activities, an experiment simulator including the spacecraft and its environment is required. The STOC simulator will support the validation of the LPF Technology Package (LTP) run procedures and the data analysis activities. It will execute the run procedures generated by means of the MOIS tool ([3]) and provide measurements to the LPF Data Analysis tools. The STOC simulator will also be used for the training of STOC personnel and to incrementally update the modelling of the LPF spacecraft and in particular of its LTP experiment using flight data.

The reused simulators

Also with the expectation to reduce the cost of the simulator development, it has been decided to base the STOC simulator on two other simulators that are already available in the LPF project. The Drag Free and Attitude Control System (DFACS) is a Windows Matlab/Simulink based simulator for the analysis of the DFACS performance. It has served as a prototype for the onboard control algorithms.

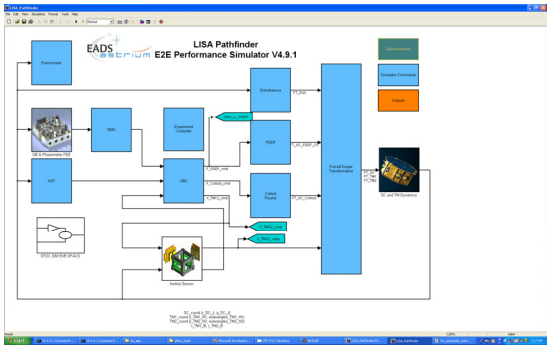

Fig. 1. The DFACS Simulator.

The Software Validation Facility (SVF) is part of the Astrium Model-based Development and Verification Environment (MDVE), where it is used for the integration and debugging of the onboard software.

The reuse of the DFACS and SVF facilities is the main constraint in the development of the STOC simulator. The DFACS and SVF provide two alternative simulation strategies, which are highly complementary in terms of efficiency and accuracy of the results. The main intent of the STOC simulator is to provide, in a unique framework, the seamless integration of the underlying simulation strategies, together with common simulation services. Those services include: recording, monitoring, injection of telecommand (TC) sequences, injection of external stimuli, and definition of parameters of the simulation models.

These simulation services necessarily rely on different mechanisms provided by the DFACS and the SVF, but it is the objective of the STOC simulator to hide as much as possible to the user the different implementation details provided by the two systems. The inputs to each service (e.g. variables to be recorded, initialization of parameters in the simulation models, etc.) are provided by the user defining a set of artefacts by means of a common user-friendly Man Machine Interface (MMI).

A detailed description of the LPF STOC Simulator can be found in [2]. The remainder of this section highlights the main issues found in reusing the DFACS simulator.

The Coupling problem

Already before the Preliminary Design Review of the LPF STOC Simulator (held in April 2008), it was clear that, although the DFACS status was rather stable, modifications from ASTRIUM could be expected. In order to limit the impact of configuration control issues, the DFACS extensions needed for the STOC purposes (e.g. monitoring,

recording, etc.) are decoupled, as much as possible, from the ASTRIUM line of development. Those extensions are implemented as separate libraries, directly depending on the Matlab/Simulink layer. The Software Reuse File document is carefully maintained, so that new versions of the ASTRIUM DFACS can be safely and efficiently incorporated.

Furthermore, the interface between the STOC Simulator MMI and the DFACS environment is mainly based on intermediate text files. This increases the observability, and minimizes the coupling between the two systems.

The Scope problem: commanding the system

Given the more limited scope of the DFACS simulator with respect to the operational context, an important factor to the success of the project was the development of an appropriate DFACS adapter, allowing mapping, when needed, the spacecraft TCs to their counterparts in the design simulator. The mapping itself is defined in an XML file. The maintenance of this XML file is a critical task. Specific tools (see Fig. 2) have been designed and embedded into the STOC Simulator, allowing:

- Viewing the XML mapping by means of a MMI.
- Showing a report of the dependencies between different TCs, as they may be implicitly specified in the XML mapping.
- Validating the XML mapping w.r.t. a given version of the TM/TC database.

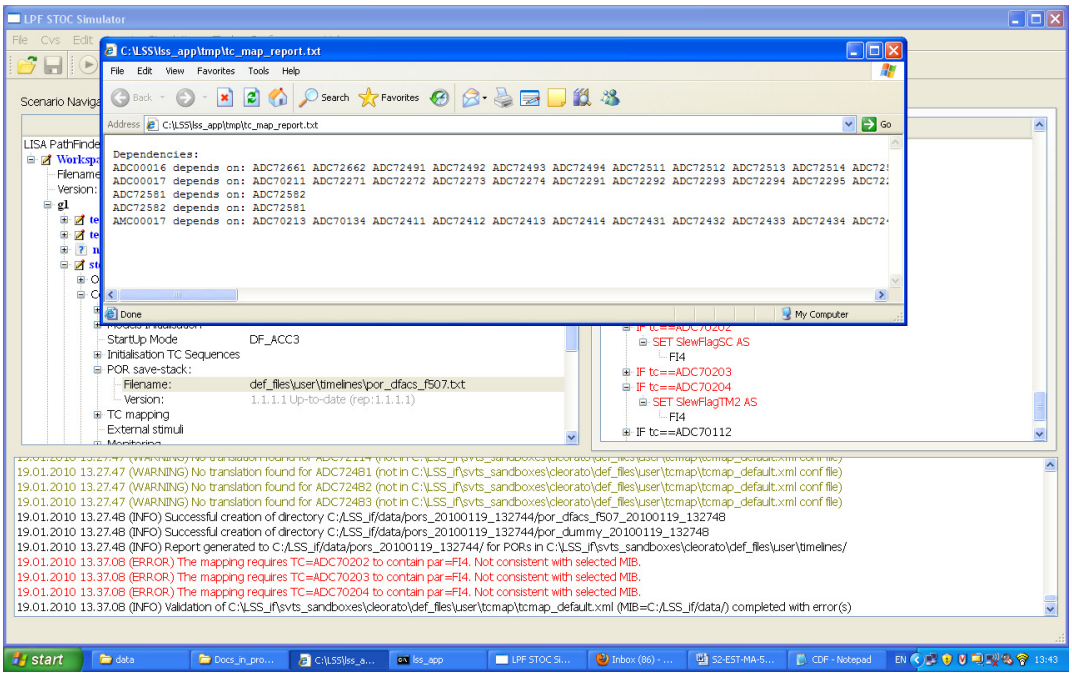

Fig. 2. The LPF tool validating the DFACS TC mapping.

More on the Scope problem: initializing the system

Given the different nature of the two simulators, the mechanism to initialize the simulation variables is very different in the DFACS and in the SVF environment. In the SVF, data integrity is ensured by a dedicated simulator database, defining the relevant properties of the simulation variables. For example, relevant properties are: default value, dimension, range, etc. The documentation provided by ASTRIUM for the numerical models is consistent with the format and naming conventions specified in the simulator database.

On the other hand, the initialization on the DFACS is based on Matlab scripts. The association with SVF variables is not obvious, because:

- The mapping is not defined in the simulator database: the simulator database only “knows” the SVF simulator; it “does not know” the DFACS simulator.
- Specific conventions and rules have not been enforced in the LPF project since the beginning of the DFACS development.

Nonetheless, in the LPF project, the mapping is made available by ASTRIUM, in the form of additional text files. These text files are exploited by an appropriate DFACS adapter, which can derive the actual initializations to be performed in the Matlab environment. This process is transparent to the user. The user is only requested to specify the initialization in

a common MMI (see Fig. 3), using essentially the same format and naming conventions specified in the simulator database, and in the documentation provided by ASTRIUM for the numerical models.

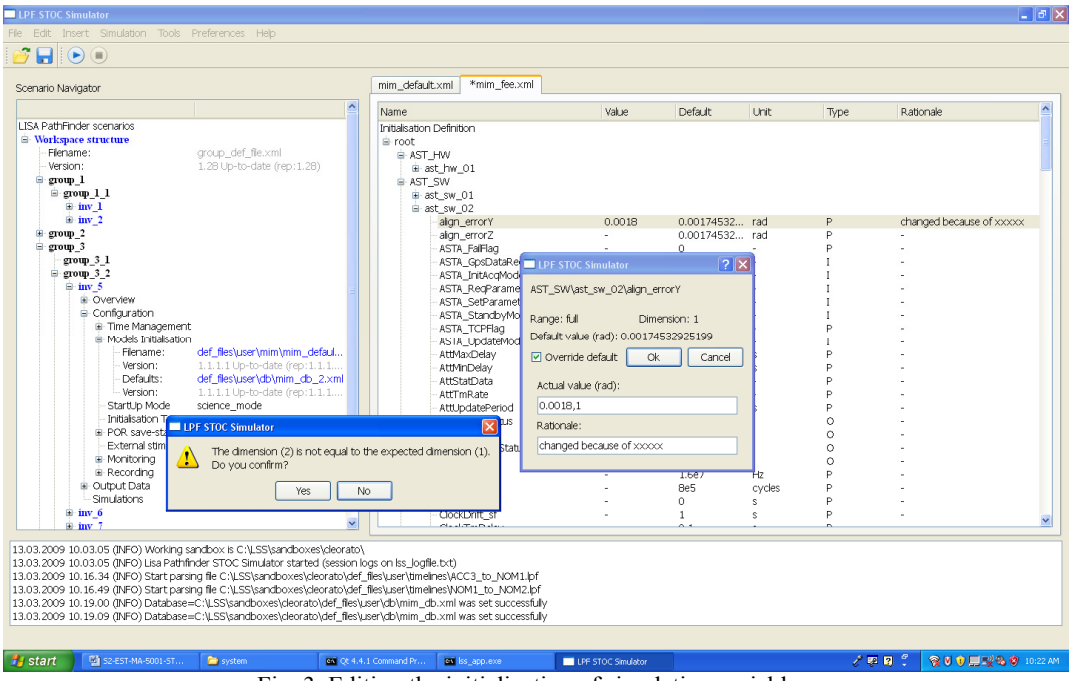

Fig. 3. Editing the initialization of simulation variables.

The Restore problem

The LPF STOC simulator user requirements included the capability of defining "breakpoints", intended as the capability of saving/restoring the simulation status. From a technical point of view, the complete implementation, verification, and validation of this requirement introduce significant complexity on the development.

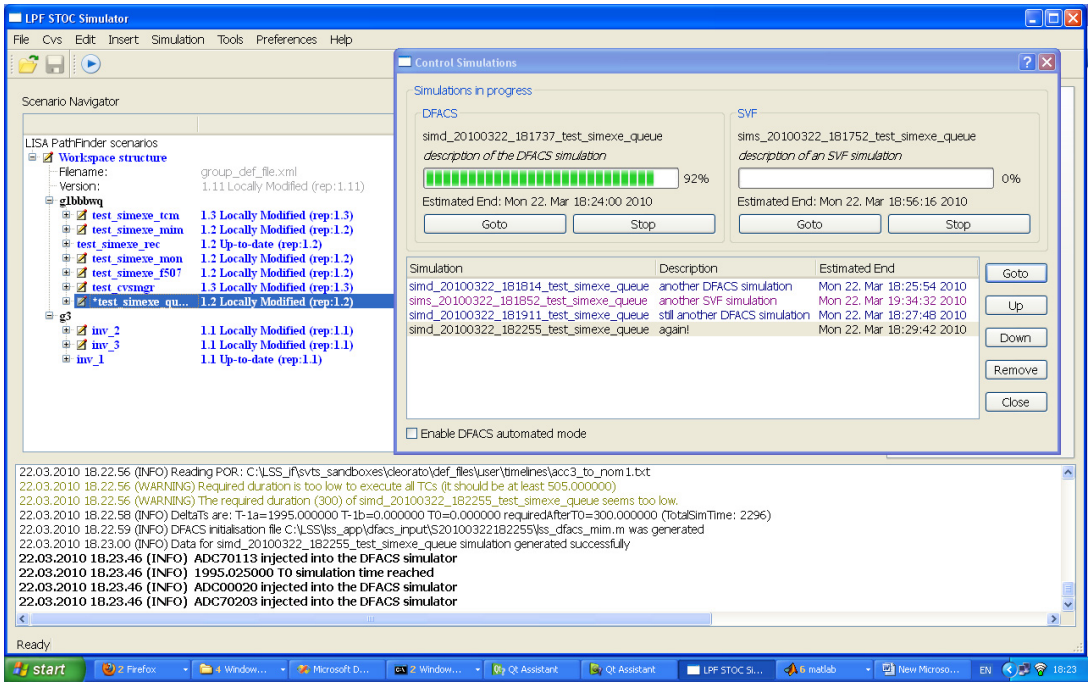

Fig. 4. Executing simulations in batch mode.

However, after more detailed discussions with the final users, it was concluded that only a limited number of breakpoints are needed by the STOC. Furthermore, given the high speed factor of the DFACS simulator (25x real-time),

the time to reach each any of the identified required breakpoints from the default initial DFACS status is actually negligible.

Therefore, instead than implementing the full save/restore capability, a light approach has been chosen:

- Special initialization sequences have been identified. The user can easily select them, and the sequence is fully simulated, any time it is required to reach the conditions corresponding to the required "breakpoint".
- Because it is fully acceptable for the STOC to analyze the simulation results off-line, the STOC simulator allows queuing simulations, executing them in batch mode (see Fig. 4).
- If required, the STOC simulator may be easily enhanced, so that different DFACS simulations can be executed in parallel, exploiting the different cores of the STOC Simulator machine.

THE ATV OPERATIONAL SIMULATOR

The development of the ATV Ground Control Simulator (AGCS) has leveraged from the other Test Facilities which had been developed for the ATV, e.g.:

- The Functional Simulation Facility (FSF).
- The Software Validation Facility (SVF).

The usage of appropriate Simulink programming standards has allowed meeting the required real-time performances. Numerical models could be extended, e.g. implementing the specific failures required to train the ATV Control Center operators.

However, it shall be noted that the ATV real-time Test Facilities include the design simulators of major hardware subsystems, including:

- The Solar Array subsystem.
- The Propulsion subsystem.
- The Docking and Refueling subsystem.

Because the design simulators were developed in textual languages (C and Fortran), an additional effort has been required in order to incorporate them into the Matlab/Simulink development chain. For this purpose, a rigorous approach was needed, in order to:

- Handle the interfaces of the reused simulators towards the rest of the simulation infrastructure, e.g. to define how the reused simulator shall be commanded and initialized.
- Define an automated testing environment, to reduce the testing overhead, in case of deliveries of new versions of the design simulators.
- Correctly implement the save/restore capability.

The analysis of all these three aspects has contributed to the definition of the overall design. It has also contributed to the definition of appropriate and consistent procedures and tools. Further details are provided in [1].

Interestingly enough, the three aspects above closely mirror the three key issues identified in the previous section, regarding the reuse of the LPF DFACS simulator:

- Identify the scope of the simulator, in terms of its interfaces (e.g. commanding and initialization of the reused system) with respect to the overall operational context.
- Identify and minimize the coupling of the design simulator with the overall operational infrastructure, i.e.: define and maintain the (possibly automated) procedure to incorporate the changes in the design simulator into the operational system.
- Analyze the rationale for the save/restore requirement, and identify a suitable and long-term viable strategy, in order to satisfy the actual user needs.

THE METHODOLOGY

This section simply summarizes the natural generalization of the experiences reported above.

In order to assess, for a given project, the actual feasibility and cost of the reuse of design simulators in an operational context, it is essential to formalize and analyze at least three key views of the design simulators. These views respectively focus on:

- The scope of the reused simulators.
- The coupling of the reused simulators with respect to the overall operational simulator.
- The implementation of the save/restore capability, and of other capabilities specific to operational simulators.

The analysis of each one of these three aspects shall rely on the project context, including:

- The project background.
- The items already developed for the project, their relationships, and their governance.

- User requirements.

The analysis of each view will identify specific needs in terms of design, processes, and tools. Once this analysis has been completed, it is therefore essential to review the results under the global perspective, in order to achieve a consistent design, as well as consistent processes and tools.

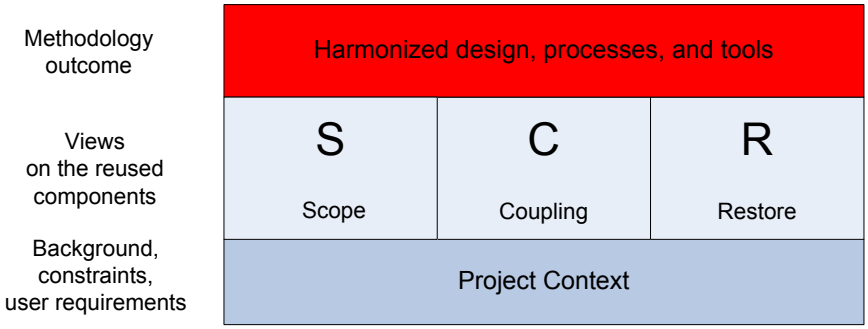

Fig. 5. Reuse of design simulators in operational contexts: the Global view.

The Scope view

Operational simulators have a different scope than design simulators. An effective reuse requires the development of adapter layers, e.g.:

- To map spacecraft TeleCommands (TCs) to their counterparts in the design simulator.
- To initialize the status of the design simulator.

It is essential that the required adapter layers are maintained up-to launch and beyond. For this purpose, the development of appropriate auxiliary tools may also be needed. Auxiliary tools typically include Man Machine Interface (MMI) editors, tools validating the adapter layers, comparison tools with data in reference databases, etc.

It is recommended to identify, at an early stage, the boundary between the legacy systems and the new operational infrastructure, taking particular care to also ensure that any inputs required by the auxiliary tools are available and will be maintained in the long-term, according to the project context.

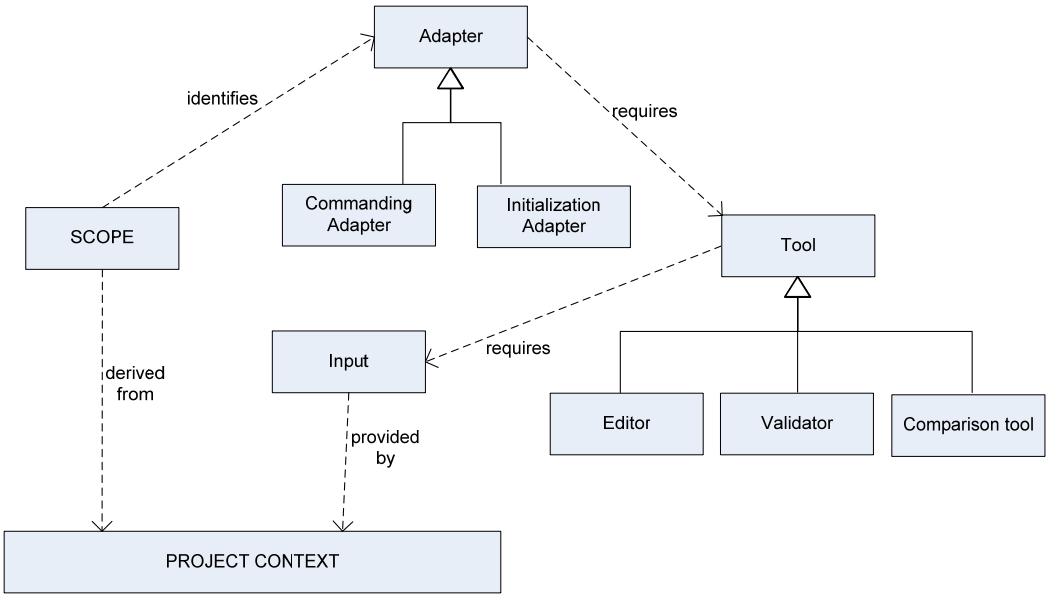

Fig. 6. The Scope view.

The Coupling view

Coupling with reused simulators shall be minimized. Taking into account the project context, the risks deriving from concurrent updates of the reused simulators shall be carefully assessed. Appropriate procedures shall be defined and maintained, in order to safely incorporate the changes into the overall operational system.

The Restore view

Specific requirements need to be considered for the development of an operational simulator:

- Modelling of specific equipment failures.
- High speed-up factor and/or the capability to save the simulation status, and restore it when requested by the users.

Modelling of specific equipment failures is not usually an issue. In case the associated development shall only be part of the operational simulator, this shall be taken into account in the Coupling view.

On the other hand, a critical aspect in operational simulators is its throughput in terms of significant simulations which can be completed in a given time interval. Throughput is a monotone function of the speed-up factor and of the level of save/restore compliancy (e.g.: 0= save/restore is not supported, 1= save/restore is fully supported). The objective is therefore to optimize the throughput, as described in Fig. 7.

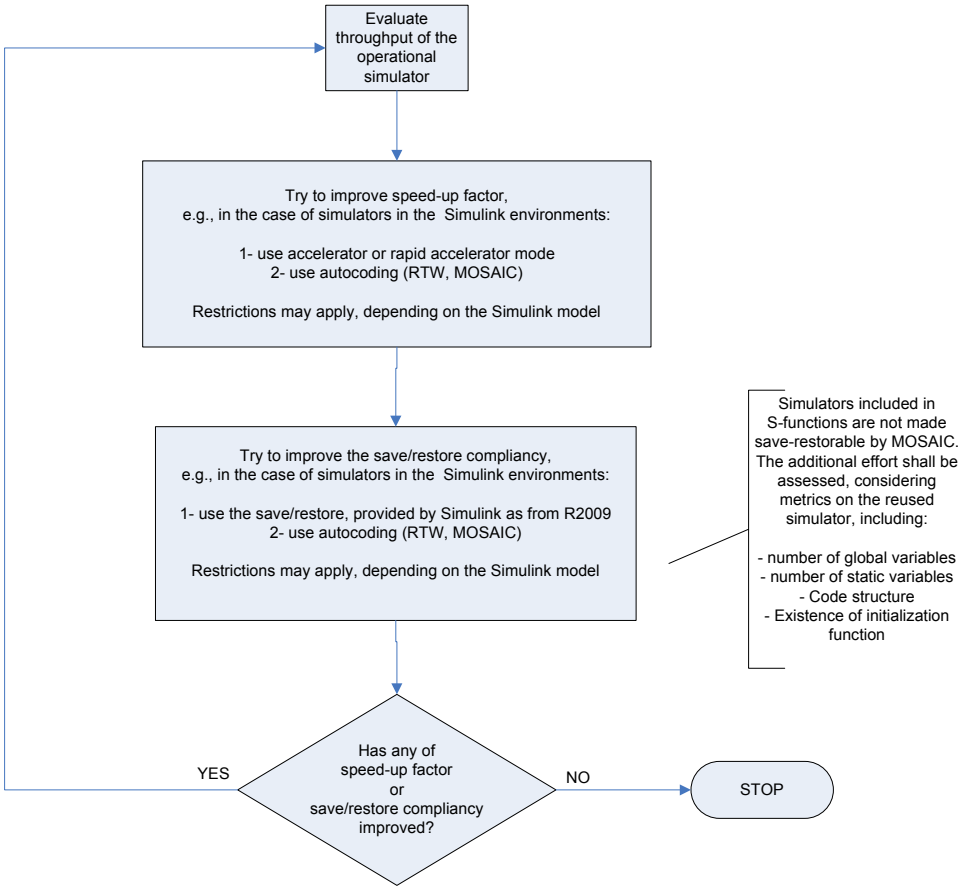

Fig. 7. Flow-chart for the Restore view

In practice, the save/restore capability is often the aspect which is more likely to determine that a given design simulator is not retained for usage in the operational context. The implementation of the save/restore compliancy on existing simulators is often a time consuming and error prone activity. Procedures and tools shall be established to ensure that the save/restore implementation is thoroughly validated, by test and by review of design, before embedding the reused design simulator into the operational run-time simulation infrastructure.

CONCLUSIONS

Reuse across the project life-cycle has been highly beneficial in the ATV and LISA Pathfinder missions. The approach has increased the coherency between the different simulation facilities. Nonetheless, when it is intended to reuse design simulators as part of the operational infrastructure, potential pitfalls exist. In future missions, risks can be mitigated by applying a set of proven guidelines.

REFERENCES

[1] C. Leorato, E. Guidolotti, D. Segneri, "Reusing Legacy Code in a Simulink Environment for Real-time Industrial Simulations: Techniques and Prototype Support Tools", Proceedings of the DASIA Conference 2005, ESA SP-602 (August 2005) pp. 316, 327.

[2] P. van der Plas , C. Leorato, "The LISA Pathfinder Simulator for the Science and Technology Operations Center: Simulator reuse across the Project Life-cycle: practical experiences and Lessons Learned", Proceedings of the DASIA Conference 2010, in press.

[3] O. Camino, R. Blake, W. Heinen, "Smart 1 Scheduler – A Cost Effective Mission Scheduler compatible with SCOS2000", Proceedings of the 4th International Workshop on Planning and Scheduling for Space, ESOC, Darmstadt, 2004.

[4] A. Walsh, Q. Wijnands, N. Lindman, P. Ellsiepen, D. Segneri, H. Eisenmann, T. Steinle, "The Spacecraft Simulator Reference Architecture", Proceedings of the 11th International Workshop on Simulation & EGSE facilities for Space Programmes, ESTEC, 2010.

[5] P. Fritzen, D. Segneri, M. Pignede, "SWARMSIM - The first fully SMP2 based Simulator for ESOC", Proceedings of the 11th International Workshop on Simulation & EGSE facilities for Space Programmes, ESTEC, 2010.