SCOE SIMULATION

Pascal CONRATH[(1)], Christian ABEL[(1)]

Clemessy Switzerland AG[(1)]
Gueterstrasse 86b
4053 Basel, Switzerland
E-mail: p.conrath@clemessy.com, c.abel@clemessy.com

ABSTRACT

During the last decade Clemessy has developed, in his automotive business unit, a versatile simulation platform embedding software and hardware, which is able to simulate the electrical behaviour and interactions of organs of a car under development. On the other side, Clemessy Switzerland, Clemessy's space division, delivers EGSEs since more than fifteen years to the European space industry. The convergence of technologies in both business units, and the increasing needs of optimised validation, drove Clemessy Switzerland to develop a simulation platform, which is used for its product development and validation, and which can have many further applications in the future.
The tool itself will be improved in the near future to cope with coming extensive needs to demonstrate compliance of delivered products which ultimately are connected to flight hardware of highest value.

INTRODUCTION

The development of an Electrical Ground Support Equipment dedicated to a scientific satellite is a project which lasts between one and two years depending on the complexity and the number of subsystems to develop and on the procurement time of some long lead items.

Even if they are mainly based on "commercial off the shelf" products, due to the integration of different kinds of technologies, EGSE systems embed often dedicated hardware and software developments, which need time to be realised and tested.

In a general context of reduced budgets and schedule shrinking, where re-use of existing developments is not always possible, the use of simulation techniques can be justified, in order to overcome the problem of lead times of missing items, enabling some schedule optimizations during the validation phases.

TYPICAL SCOE IMPLEMENTATION

A SCOE [1] (Special Check-Out Equipment) is a system which essentially simulates parts of a satellite during the AIT activities, allowing :

- To perform tests on subsystems of the spacecraft units by providing power, simulating signals/stimuli and communications (e.g. Satellite instruments tests)
- To support integration activities and integration tests by simulating partly or completely the missing subsystems (e.g. solar array simulation, battery simulation, pyro simulation, etc.)

In general the architecture of the SCOE is the following :

- A **SCOE controller** (computer), with a local MMI to drive the system. The SCOE controller is connected to the customer's CSS (Central Check-out System). This controller manages all the front ends of the SCOE.
- Various **front ends**, which are interfaced to the satellite during the integration/tests. These interfaces are of various types depending on the SCOE type :
    - Power : Power Supplies, Electronic Loads, Protection Devices, LCLs
    - Communication : MIL-STD-1553, SpaceWire, CAN, Serial communications,
    - Measurement : Analog, temperature, pulse acquisitions, etc.
    - Signals/Switches : Digital IO, relays switches, High Power Commands, etc.

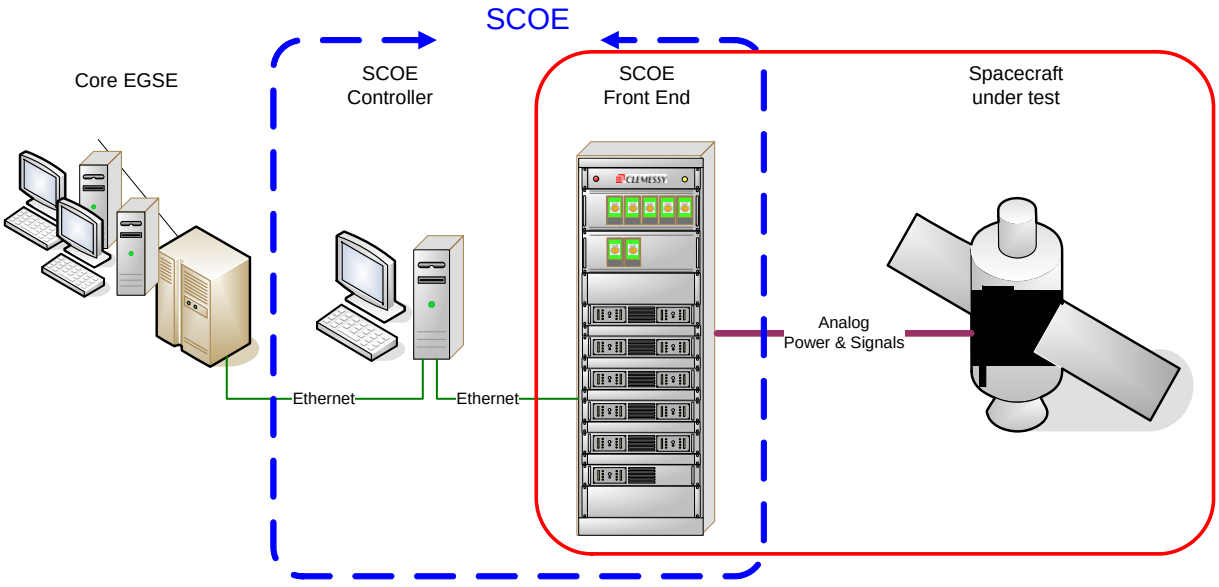

Fig. 1 Overview of a typical SCOE

The SCOE controller is based on a re-used software where all interfaces to the front-ends are already existing. Likewise, most of the CCS interface protocols are already implemented in the controller software.

The development of the application software for a dedicated SCOE consists of :

- a combination of re-used functions and the development of SCOE specific functions, required by the customer
- the development of a customized MMI dedicated to the programme
- the use or adaptation of a CCS interface protocol

USES AND BENEFITS OF SCOE SIMULATION

SCOE simulation is useful in the following cases :

- **Early validation of SCOE controller software**
  The validation of the Controller software, which drives the SCOE front-ends can normally be performed only when all the SCOE items are available, fully assembled in the racks and when all individual hardware items have already passed the unit tests. In general this activity takes place some weeks before the factory acceptance, and can impact the final schedule, because testing activities are on the critical path.
  The support of a simulator for the software validation clearly allows to cover most of the software functional validation far in advance, with respect to the availability of the full hardware. This helps to limit most of the software validation activities, after hardware integration, to “interface solving” issues, more than to functional issues, which can save a couple of weeks on the overall schedule and to minimize the impact on the schedule of possible software issues.
- **Increase test coverage possibilities**
  Some test can be difficult or even impossible to implement with real hardware, because the test setups may change the system configuration in a different way than the delivered product or that they could even destroy SCOE items (blow a fuse, stress a device by overvoltage, etc.). By using a simulator, these tests setups can be easily made, and the assessment of the according risks (confirmation of FMECA assumptions) can be made without any hazardous impact on the system.
  Furthermore, with simulations embedding also satellite models, possible hazards on the interfaces between the SCOE and the satellite could be assessed, where only assumption based demonstrations where used formerly.
- **Support the customer’s test preparation activities**
  After the delivery of the SCOE, the first task of the satellite integrator is to integrate the SCOE in its EGSE configuration. The main activities being the interfacing of the SCOE with the CCS, with validation of the communication between both systems and development/tuning of the CCS MMI and population of the database with the SCOE mnemonics.
  The early delivery of a SCOE controller together with the SCOE simulator allows the satellite integrator to perform these activities in advance, before full SCOE delivery. The testing teams are also able, with the simulator, to develop and test their application procedures without having yet the final SCOE hardware.
  Globally at customer level a lot of tasks can be performed in advance saving time or leaving it available for other activities.

- **Support the customer's advanced or remote training**
  No need of complete hardware to support the training of the customers. With the simulator, the basic training on the SCOE can be performed before the hardware is delivered, at any time and any place.
- **Allow the SCOE provider to perform fault diagnosis**
  After delivery of the SCOE to the customer, it can be difficult in some cases for the SCOE provider to provide support for diagnosis, since remote operation of the SCOE is to be avoided for safety and security reasons. Therefore, the replay of the failing test scenario on a simulator, can be helpful to find the root cause of the failure before any on-site intervention.

BACKGROUND OF THE SCOE SIMULATOR DEVELOPMENT

The development of the SCOE simulator is based on research performed by Clemessy back 8 years ago in the automotive area. At this time started a company project called TestInView, which aim was to provide to automobile designers and integrators a complete hardware and software tool able to simulate electrical parts of a vehicle in development. This simulation being not a software simulation but a physical simulation based on COTS items from National Instruments driven by a software suite developed in our company [2].

This solution, which can be customized to various applications, is mostly used during the design phase and prototyping phase of a vehicle, when at the beginning of the phase only few hardware components are available and when the integration already starts. The TiV solution allows to simulate the missing vehicle electrical organs by front-ends which are electrically bound to other organs (real or simulated), this allows to close the loop and to start full functional tests at electrical subsystem level. As soon as the real hardware item becomes available it will replace the simulated prototype in the electrical breadboard of the car in development. The tests will then continue until all simulators are replaced with real organs.

The development by Clemessy Switzerland of the SCOE simulator is based on this former developed TiV tool, with dedicated customizations for the interfacing with the already existing SCOE controller platform of the company.

WHAT CAN BE SIMULATED IN A SCOE ?

The aim of SCOE simulation is to support the SCOE controller software development by simulating all the SCOE low level items, which are ultimately in direct interface with the spacecraft.

Simulation of the SCOE Front-Ends

For a SCOE, the front-end models include the description of following items :

- Power Supplies commanding, status and behaviour : IV curve for rectangular (fixed mode) power supplies, three points model (VOC, ISC, VMP/IMP) for Solar Array Simulators

- Electronic loads commanding, status and behaviour : Constant Current, Voltage , Resistance or Power
- Protection devices commanding, status and behaviour
- TM/TC simulation
- Digital and Analog signal simulation
- Etc.

The simulator model takes also into account the functional interactions of all the instruments that are connected together.

Simulation including the Spacecraft electrical behaviour

Since the aim of the simulator is mainly to verify the functional behaviour of the SCOE and not to perform a detailed electric circuit simulation ("Spice" like), the device under test models are limited to simple models. Real time simulation is in the range of 100ms, which is far enough at SCOE level validation.

Some examples of simulations that can be performed on a Power SCOE :

- **Power consumption of the satellite**
  The model describes the spacecraft loads which are connected to the SCOE power outlets (solar array regulator, loads connected to the main bus, battery regulator). In our model the loads are represented with resistors, not taking into account the capacitance and the inductance.
- **Switching of loads on the satellite**
  This kind of simulation enables to test the behaviour of the SCOE facing changing load levels on the spacecraft, and verify particularly the performance of the electrical devices and help to focus on protections (nominal and redundant) embedded on the SCOE, which aim to protect the satellite.
- **Simulation of satellite on board values**
  One of the essential tasks of the SCOE is the monitoring of spacecraft physical values (on board voltages, currents, temperatures). All these values are simulated in the model and fed back to the SCOE controller .
- **Simulation of satellite communication**
  The models also provide possibilities to perform communication simulation, which remains at packet level, enabling to check the complete commanding chain from the CCS to the spacecraft and the way back.
- **Simulation of failures**
  One of the most interesting feature of simulators in general, which is used here for SCOE simulator is the ability to inject failures at the model level (over-voltage, over-temperature, wrong data packets, etc.), in order to check the reactions of the SCOE at all levels of the controlling software up to the CCS. This helps to improve, during the system tuning activities the handling and display of errors, to avoid any possible misleading information to the user, and to finally check that the system remains in a secure configuration after a failure (e.g. latched status).

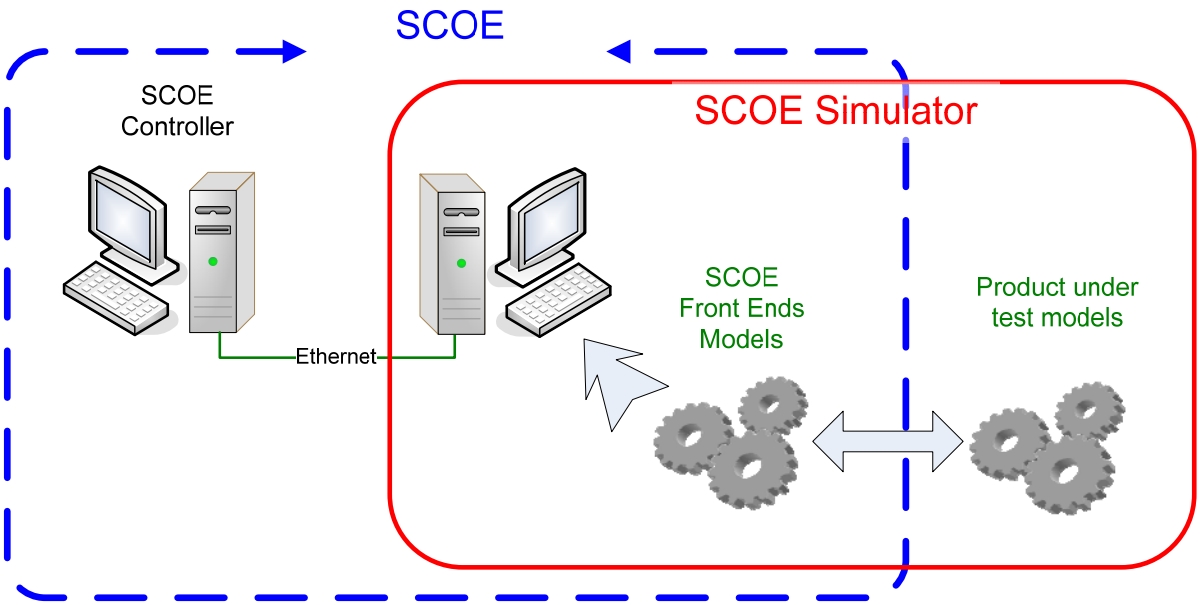

Fig. 2 What can be simulated in a SCOE ?

The proposed simulation is a functional simulation which allows to simulate changing levels, and is not focused on the high speed transient events (due to e.g. capacitive or inductive components in the circuit). The simulation is used to validate the integration of electrical devices at their functional level.

KEY TECHNICAL POINTS TO THE SIMULATION

Establish the interfaces between the SCOE controller and the simulator

At SCOE level, all hardware elements to simulate are usually driven by the SCOE controller through communication media (Ethernet, RS232, RS485, GPIB). Therefore, the simulation environment needs first to handle communication protocols and interpretation of commands. For each type of instrument (power supply, electronic load, acquisition system, protection system, etc) its communication is described as a functional grammar in the simulator.

Because today all instruments are directly or indirectly able to communicate over an Ethernet link, it is easy to replace an instrument by a computer having an Ethernet link and acting as an instrument "server", meaning it behaves like the real instrument, communicating with the SCOE the same way as the real instrument. By using various ports on the server, it is easy to simulate multiple instruments on one SCOE simulator, each virtual instrument being connected to a different Ethernet port.

Virtual instruments server

The simulator then acts as a "virtual instruments" server, where each real instrument is replaced by a software object, to which the corresponding communication channel to the SCOE controller is bound.

The SCOE simulator is composed of software objects which are instantiated as many times as there are instances of the real instruments. All these instances will run in parallel and in real time, to send answers to the SCOE Controller requests, similar to the real instruments.

For each "virtual instrument", a grammar has been defined, which allows to manage the communication protocol of the instrument and which allows to interpret the commands sent in the packets. A behaviour model is also described for each received command of the "virtual instrument".

Synchronicity between SCOE controller and SCOE simulator

Even if the system is not designed to perform "real world" simulation, the timing and responsiveness of the system must not be ignored. The SCOE controller sends commands to the instruments at a certain pace, expects feed-back from those devices and can generate time-out events in some cases. The simulator must therefore fit to the timing contingencies defined by the controller, otherwise, each single command sent from the controller to the simulator could result in a time-out.

The "real time" aspect is therefore defined with respect to the cycle time of the SCOE controller and the reactions times of the real devices controlled by it.

SIMULATION TOOLS AND FUTURE EVOLUTIONS

The original SCOE simulator has been developed in C# programming language, the same as the SCOE controller itself. Main ideas where to use as much as possible all the communication drivers already written since they are symmetrical between the controller and the simulator.

The model part is also written in C# because the major part of the simulated instruments are well known and can be re-used from project to project without need to change their behaviour model. On the existing SCOE simulator, the satellite part of the electrical model is for the moment simple enough that C# programming is sufficient.

Evolutions to third parties models

Future evolutions are foreseen to interface third party simulation models like MATLAB/Simulink® in order to simplify the integration of models developed with these tools, the interface part to the SCOE controller remaining drivers written in C#. The use of standard modelization tools will also potentially widen the available models base and help users aware of MATLAB/Simulink® to develop their own models.

Improvement of the models

Future work will be to improve the representativeness of the models in terms of accuracy and reaction (e.g. possibility to simulate electrical transients)

Improvement of simulation parameters and test sequences

Once the simulation is operational the remaining and time consuming task, is to define the parameters and test sequences with which the simulation shall run. The injection of failures or even sequence of failures (which can easily happen in stress situations) into a model, need additional tools still to be developed to support the generation of those scenarios.

CONCLUSION

The developed tool is for the moment a preliminary version, which is very promising with respect to the objectives to improve our SCOEs development cycle. A lot of work is still to be done to go towards simulation standards to simplify the re-use of the simulator itself from one SCOE project to the other. An additional effort will also be done to package the tool as a deliverable to our customers, but this is a further step!

REFERENCES

[1] P. Conrath, C. Abel, “From one shot EGSE to generic EGSE”, Proceedings of ESPC 2011, European Space Agency

[2] G. Sauner, P. Roth, “TestInView”, NIWeek 2012, unpublished