The EGSE for ESA VEGA launcher:

A new approach to combine versatility and performances

Massimiliano Mazza[1], Giulio Troso[2], Simone Dionisi[3], Andrea Costantino[2], Enrico angioli[2]

[1]*Vitrociset Belgium, permanent establishment in the Netherlands*
'sGravendijkseweg, 53 – 2201CZ Noordwijk (NL)
Email: m.mazza@vitrociset.be

[2]*Vitrociset S.p.A.*
Via Tiburtina, 1020 - 00156 Roma (Italy)
Email: g.troso@vitrociset.it
andrea.costantino@vitrociset.it
e.angioli@vitrociset.it

[3]*former Vitrociset Belgium, permanent establishment in Germany.*
Lise-Meitner-Strasse 10, 64293 Darmstadt (Germany)
Email: simone.dionisi@gmail.com

1 INTRODUCTION

The aim of this paper is to demonstrate how it has been possible to develop an innovative and performing testing suite of applications keeping adherence to applicable ECSS standards and basing the architecture on the ESTEC EMCS Reference Facility, showing the actuality and the adequateness of its design to real applications and not only to theoretical concepts.

The description of our design wants as well to show Vitrociset's approach to AIT quality assurance enforcement as an added value to the whole development and testing process.

2 DESIGN BACKGROUND

The idea behind the design of the VEGA EGSE is the reuse of the architecture developed by ESA/ESTEC in the definition of the so called EMCS Reference Facility (where EMCS stands for EGSE and Mission Control System). Besides some other partial deployments of similar but limited architectures (PROBA 2 EMCS), the VEGA EGSE made use of the full suite of components. The rationale of this choice is given by the high devotion to standards this architecture provides and its embedded vocation to modularity and scalability.

The goal to achieve was an EGSE with stringent requirements in terms of safety and performances, scalable and modular such to be able coping as well with sub-units testing as well with the complete assembled launcher (three propulsion stages plus upper composite and fairing). Another reason for the choice was the envisaged advantage in exploiting all possible commonalities between the EGSE and the Mission Control System mechanisms.

The ERF intrinsically responds to the idea behind those requirements. Its architecture is a customisation of the standard ESOC MCS (SCOS 2000 r3.1), with extensions to support an EGSE protocol (to manage SCOES) and to be driven by a scripting language. By this means the DB ICD, the way TC are formatted and created, the way the TM is decommutated and monitored are de facto reused from the MCS. This approach provides many advantages like:

1. reuse of components and applications in 2 separate scopes of the project with cost savings
2. preliminary validation of the MCS suite through use and troubleshoot of the EGSE
3. validation of the very same Mission Database

The scheme of the EMCS concept is depicted in Fig. 1 courtesy of ESA.

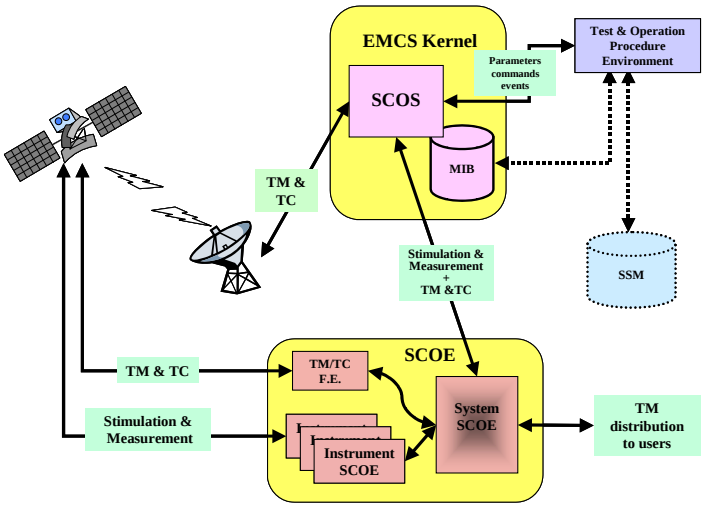

Fig. 1: EMCS Reference Facility architecture

3 DESIGN DESCRIPTION

A first embryonic deployment on a real project of the ERF architecture on a mission has been the PROBA mission. The number of components involved and the performance requirement did not make the expected sensation therefore the return in image for the ERF has been less than hoped.

The VEGA EGSE ambition was instead pursued with an important deployment of HW resources able to host and run a massive suite of applications intended to tackle all the performance, reliability and quality constraints required.

As first the function of test environments production (TCS) was decoupled from the execution (TES) to enhance the productivity by parallelising, and to improve the safety by physically preventing unauthorised test environments to be executed on the qualification or flight HW. The Test Configuration stations have been provided therefore with SCOE simulators to be able running the test sequences in debug and validation mode. A versioned repository with strict rules has been interposed between the TCS and the TES to allow keeping track of all editing and execution transactions to allow only execution of validated and approved test environments.

3.1 Test Configuration subsystem

The Test Configuration System software suite includes a DB editor (with wizards guiding through the data population and browsing), an integrated Space System Model and PLUTO editor, a MIMIC editor, a Model Integrity Tool (MIT) and a Session Manager. This suite of applications concurs in creating and maintaining Test Environments. These objects are in a sense atomic, meaning not intended to be divided and handled in parts. The Test Environment handling as a whole is one of the envisaged means to preserve consistency and correctness. Every editing is tracked and forced to consistency wherever possible, otherwise post-processed and cross checked by the MIT. The Session Manager is the gateway to the Repository. It takes care of collecting all the information building up the Environment and submitting it as a unique object to the versioning system. The Session Manager provides as well the means to tag environments for execution. Only environments tagged through a defined cycle of approval can be deployed on the TES and be executed on the real HW.

A special mention in the TCS should be given to the TPE (VEGA EGSE version of the more common ASE3). TPE stands for Test Preparation and Execution. It is composed by different modules that allow the tool to provide various functionalities according to the environment where it runs. At the same time it embeds common functions, due to the wide communalities between the TCS and HLCS environments in some subset of tasks.

It integrates an interface (read / write) to the Space System Model (SSM, ECSS-E70-31) within the framework of the Procedure Language for Users in Test and Operations (PLUTO, ECSS-E70-32). The Space System Model is used within PLUTO as a mean for capturing mission knowledge for AIT and operations. The knowledge is used by EMCS

applications in order to interact with the space system and to process the dynamic data that has been exchanged with it (i.e. telemetry, telecommands, etc.). In particular with the creation of different modules for each UUT it is possible for the TPE to:

- Handle the reporting data and events coming from different SCOEs and consumed by running procedures
- send activities requested by running procedures to be executed by the SCOEs or forwarded to the launcher

Used within TCS, it allows populating the SSM and consequently the activities, the test procedures and the test sessions and importing / exporting them. It delegates these actions to two integrated modules: the SSM Editor (SED) and the Test Integrated Development Environment (TIDE, a proficient procedure creation tool composed by a text editor and an easier graphical editor). The SSM handler foresees as well the possibility to import (and consequently export) activities, parameters, procedures, as well as complete branches of information, with also all the MIB data associated (OOL thresholds, calibrations, actions, packet characteristics, commands etc.).

Once created the test procedures it allows also running the tests against the simulators in the very same process they are meant to follow in the execution environment, providing this way an excellent and safe debugging tool.

The TCS workflow is depicted in the following schema Fig. 2 where the numbers on the connectors indicate the progress in the sequences of events. The picture makes evident how everything rotates around the Test Environment concept.

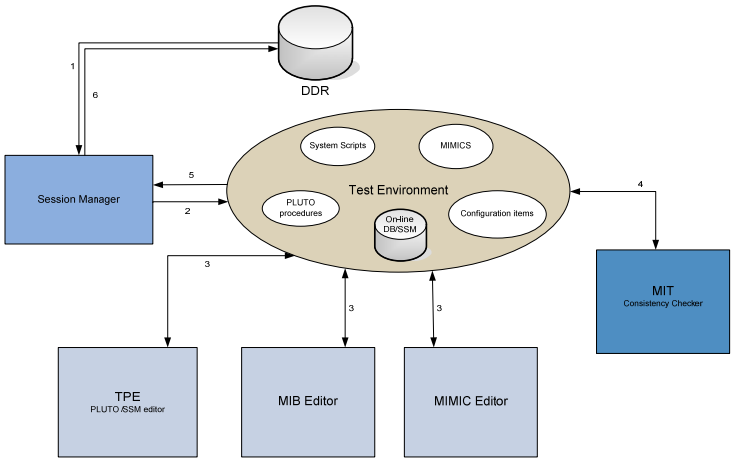

Fig. 2: TCS workflow

3.2 Test Execution subsystem

Finalised and approved Test Environments can then be deployed on the Test Execution Environment. The TES is the EGSE section aimed at executing test session, with capabilities of performing immediate data analysis and real time parameters monitoring. The TES is composed of two main elements: the CCS (Central Checkout System), also known as HLCS (High Level Control System), and LLCS (Low Level Control System). This is a scalable and modular chain whose composition and dimension can be decided accordingly with Test Environment's constraints.

Every time a Test Environment is checked out an EGSE Configuration Manager is invoked to be able targeting:

1. on which EGSE chain it is going to run (chains composition is predefined but every hostname can be overridden)
2. how many and which non mandatory components will be attached to the chain for the test run.

According to the choices the Session Manager deploys the selected Test Environment on the target hosts spreading to the right location configuration and data items after aliveness and readiness test has been performed on all expected machines. The very same process then commands importing configuration and test data in all the applications in the correct sequence, where this concept apply, and finally starts all the EGSE suite of applications. All of this automation in the VEGA EGSE context is not only a "very nice to have" but an unavoidable need driven by the constraint that

SCOEs might be close to the UUT in not accessible or potentially unsafe locations up to 4Km far from the TES stations.

The HLCS or CCS is the subsystem where the ERF is deployed. Therefore it is composed with the TPE (in its execution environment setup), the TM/TC core (SCOS-2000 r3.1) and the EGSE Router plus a dedicated component, the Commands Dispatcher, custom made to support non-PUS commanding protocols. Where the setup differs from the classic ERF is:

- the porting of SCOS-2000 r3.1 to OpenSuSE 10.3 (from SuSE Professional 8.2) to better support newer HW
- the split of SCOS tasks on an inedited client / multi-server configuration to maximise the performances of monitoring and archiving of TM versus the huge volume of data received.
- the implementation of fast commanding drivers from TPE directly to the LLCS to satisfy fast reaction loop requirements.

The HLCS is dedicated to the monitoring and control of data, data storage and interface to the LLCS.

In short the HLCS provides:

1. Monitoring data processing coming from specific SCOEs (i.e. 1553, Wiring and TM front end)
2. Archiving of all data generated by the HLCS and TPE (i.e. test events)
3. Commanding interfaces to the specific SCOEs (i.e. 1553 and TM front end) and LLCS Wiring
4. Supplies the PPS with test results data stored for the off-line Post Processing Analysis
5. Set of MMI providing the required type of displays allowing the EGSE operator to monitor and control the EGSE at HLCS level (system monitoring)
6. Test Session execution
7. Test environment/sessions management.

The HLCS has been logically structured in three top-down Service Levels:

- Service Level 3 (SL3): includes all components interfacing the users to the real monitoring and control system and in charge to prepare and execute the test environment;
- Service Level 2 (SL2): includes data repositories and all components necessary for the monitoring and control environment;
- Service Level 1 (SL1): includes the lower level components of the HLCS interfacing the monitoring and control environment to the LLCS components

The SW architecture of the HLCS is shown in following Fig. 3

The core of the HLCS is built around a SCOS-2000 installation distributed logically grouping macro functions on separate servers. User workstations instead host client installations of SCOS where the displays (TM, EV…) can be executed.

The HLCS is a client–server architecture distributed among several independent computers that are known to each other only through a network whereas the server role is managed on the HLCS environment by the CMD-D (Command Dispatcher). It is a component whose functions can be summarised as in the following bullets:

- Management of TPE clients registration to the SCOEs;
- TPE commands routing to 1553 SCOE and LLCS Wiring;
- TPE commands Inhibition management for 1553 SCOE and LLCS Wiring;
- TPE commands priorities handling;
- Parameter overriding management

The client role is managed by the TPE system. It is responsible for the execution of procedures and schedules.

The TPE clients interact with the SCOEs via drivers that have been developed as configurable plug-ins. These drivers will be loaded only if the units are really available and have been configured for the test session.

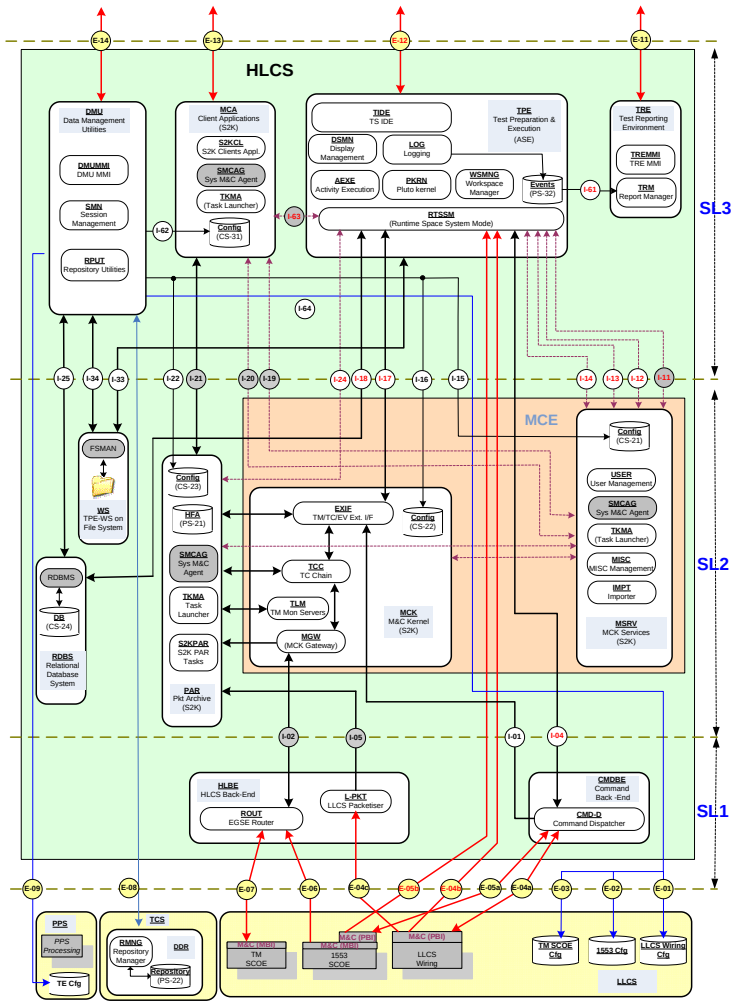

Fig. 3: HLCS SW architecture

The operator can use the GUI to execute the procedures, control the execution status, monitor the parameters values, check the variables of the procedures and control the logs generated during the test. Users with special privileges can inhibit / un-inhibit an activity or a command in order to avoid the execution of the specific command during a sensitive test session.

The TPE client comprehends another module that might be a standalone application: TRE (Test Report Environment). As already said log files are produced as a result of an execution.

Within this context, reports shall be used to provide an immediate summary of the outcome of an execution. Practically, this means that reports will allow operators to skip the analysis of the raw ASCII files by providing a user friendly view of the same data. TRE is able to generate graphical, formatted reports in PDF format.

We described also the loading procedure of the drivers plug-in on the TPE according with the configuration created by the operator. To configure the system Test Environment on the actual HW foreseen within the test session we foresee another standalone application: the Configuration Manager. It gives the user the possibility to manage the configuration of the system and the mapping of the SW functions over the available / chosen HW. Within the Editor the user can either create new units or delete them according with the availability of the UUT with a high flexibility though constrained by correctness criteria enforced by the very same tool.

The modularity of the TES is achieved by the inner capability of handling the presence of a component (some, though, are mandatory ones) in multiple instances (multi domain). This is needed for the big variety of compositions of UUT

assembly throughout the AIT lifecycle, since a single EGSE has been foreseen to be used from subsystem to launcher level, test have to be designed incrementally and, as example, cases should be avoided to deploy the full test equipment for units requiring, for instance, the hardwired SCOE only.

3.3 Low Level Control subsystem

The LLCS is composed of a set of Special Checkout Equipments (SCOE) in charge of monitoring and commanding the UUT, reacting in real-time to both nominal and anomalous events and ensuring safety conditions during the test execution, also in absence of connection with the HLCS. The performance and safety requirements make the SCOEs to be intelligent equipments remotely configured and driven by the operator from a distance of up to 4 Km and to be able to autonomously manage in safe way all critical functionalities of the UUT, operating in strict real-time. This significantly enhances safety test condition and consequently leads to a lower risk both for operators and equipments, being it able to take appropriate actions and put the launcher in safe condition.

The guideline to each SCOE design is to make equipment not only as handler of low level interface, but systems highly configurable from the TCS and able to work both under the HLCS control and in stand-alone mode. In this scenario, a key feature is the configurability of the system, allowing the users to define test sessions and SCOE behavior in terms of commands, parameters, out of limits and anomaly conditions with the related automatic reactions to be executed when such conditions occur. On the basis of the defined configuration, the LLCS is able to work in stand-alone mode, performing the following main functions:

- Acquiring data from UUT;
- Calibrating and monitoring data;
- Reacting in case of out of limits and anomaly conditions are verified;
- Answering to HLCS requests;
- Storing all acquired data and logging all activities and events for post processing analysis.

The rationale behind this implementation choice is the requirement to implement time-critical reaction loops. As a matter of fact, in the standard EGSE architecture the CCS does not guarantee reaction performances with severe time constraints; therefore it was necessary to move the execution of the time-critical reaction loops directly into the frontend equipments.

The concept of fast reaction loop involves three steps:

- measurement of signal or logic data communication (Hardwired signals, MIL-STD-1553 data, telemetry links, serial links, Ethernet links, and so on...)
- processing of the acquired data, checking values against predefined threshold and recognition of any non-nominal or out of limit condition
- execution of the reaction: generation of a physical signals or transmission of logical commands through data links within a predefined time interval.

The LLCS of VEGA EGSE system is composed of the hereafter described SCOEs.

The LLCS Wiring SCOE is devoted to the Real Time monitoring and control of the electrical signals and feeding lines (through the PS SCOE, fully controlled by the Wiring SCOE) of the VEGA launcher and to control the status of the other SCOE. This SCOE can execute preconfigured commands to out of limit conditions on any acquired parameter and can produce a safe state when any severe exception (such as one of the monitored/controlled SCOE is out of its control).

One of the key features of the Wiring SCOE is the configuration modularity. It is possible to configure the SCOE to handle additional devices: Wiring Extension, PS SCOE and Pyro SCOE. The Wiring Extension is a device that offers additional I/O signal lines used to increment the total number of hardwired channels handled by the Wiring SCOE. The physical connection of this device is implemented through fiber optic channel.

To match real-time constraints and performance requirements, the LLCS Wiring SCOE has been designed and developed using National Instruments PXI/SCXI hardware and LabVIEW Real-Time software (PHARLAP ETS Real-Time Operating System has been adopted), taking advantage from the long lasting Vitrociset experience in development of monitoring and control systems based on National Instruments products.

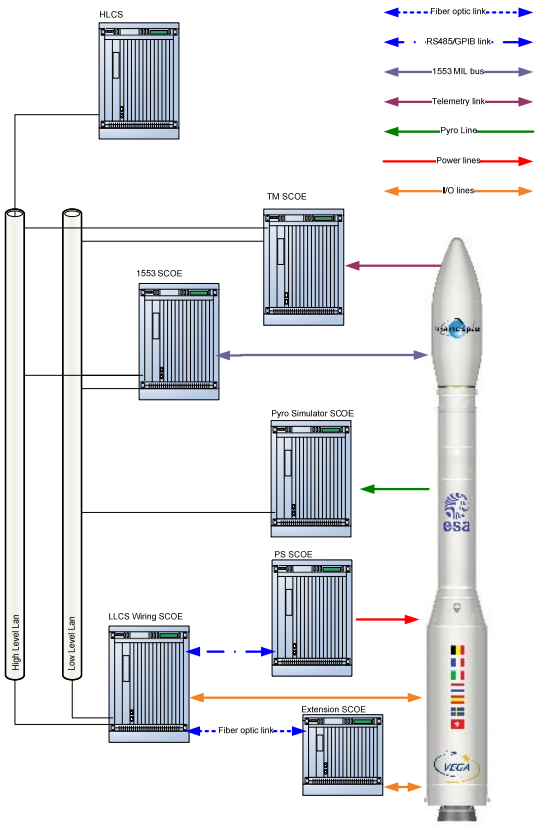

Fig. 4: LLCS decomposition

The National Instruments hardware provides high performances both in terms of signal quality and sampling rates: all the acquisition lines of the LLCS Wiring SCOE are developed using acquisition boards that offer channel-to-channel and channel-to-system isolation with up to 16 Bit coding and high sampling rates. The use of PXI boards on the same bus allows to share one clock through all the acquisition boards: this makes the system to be synchronized to one timing source and to sample all the input lines with a very low jitter (typically 50 ppm).

PS (Power Supply) SCOE acts as a remote device of the Wiring SCOE. Its main tasks are to feed the UUT through its power lines and to acquire the parameter related to each feeding line. It is composed by two unit, the first one includes the set of power supply and the control unit, the second one is composed by the distribution units that implements the electrical rails used to feed the output current to the UUT. The communication with the Wiring SCOE is implemented through a GPIB and RS485 links.

Pyro Simulator SCOE is devoted to simulate the pyro loads of the VEGA launcher. Its main task is to monitor the parameters related to each pyro line, notifying any out of limit condition detected to the Wiring SCOE. The communication with the Wiring SCOE is based onto a dedicated 100 Mbps Ethernet link.

1553 SCOE can work in stand-alone mode or controlled by the HLCS. Its main task is to interface the VEGA launcher through the standard 1553 MIL bus. It can be configured to act simultaneously as bus controller, remote terminal and bus monitor. It acquires Major and Minor frames from the bus and extracts parameters. Each parameter is processed in real time and, if any out of limit condition is detected, the 1553 SCOE notifies the anomaly to HLCS and send predefined commands on the bus; the 1553 SCOE is also able directly notify the anomaly to LLCS Wiring SCOE by means of a dedicated Ethernet, requesting it to execute Hardwired Commands.

TM SCOE is the front-end to the telemetry downlink: it acquires launcher telemetry frames through RF or baseband links, extract CCSDS packets and send them to SCOS-2000 for further processing activities.

3.4 Reacting capabilities

The reacting capability is a key feature of VEGA-EGSE. Different reaction loop mechanisms (i.e. the loop from parameters acquisition, elaboration and action sending within a specified time) have been developed to make the VEGA EGSE able to meet the challenging performance requirements and maintain a high level of configurability. The fastest loop is closed at low level, where the SCOEs can be configured to autonomously react in real time within 120 ms according to predefined actions. If higher configurability is needed, it is possible to involve TPE in the loop: by means of dedicated interfaces, TPE is able to evaluate events for from SCOE and initiating procedures as reaction. The widest access to data is achieved involving SCOS-2000 in the reaction, with interesting results in terms of time.

A benchmark on the performance achieved by the EGSE is provided on the basis of a typical scenario hereafter described:

- TM Parameters acquisition and monitoring: 200 TM parameters monitored by SCOS and based on a telemetry flow at 1 Mb/s acquired by the TM SCOE in playback mode (reading from file)
- 1553 Parameters acquisition and monitoring and frame running: 50 1553 parameters extracted and processed by the 1553 SCOE and based on a 1553 bus flow at 110 Kb/s. Parameter are distributed with 1 Hz refresh The traffic is produced by running frames and interrogating remote terminal simulated by a 1553 standard bus analyzer
- Hardwired parameter acquisition and monitoring: 46 parameters processed at 100 Hz and 227 at 10 Hz.
- Derived parameters calculation: 30 derived parameters calculated starting from the TM, 1553 and Hardwired parameters acquired by SCOS.
- Displays:
    - 50 parameters displayed in TPE AND and refreshed at 1 hertz;
    - 32 parameters displayed in SCOS AND and refreshed at 1 hertz;
    - 50 parameters displayed in SCOS Mimics and refreshed at 1 hertz;
    - 50 parameters displayed in SCOS GRD and refreshed at 1 hertz;

Performances in terms of reaction times:

Reaction loop mechanism	Reaction loop description	Reaction times
Fast Reaction Loop at LLCS Wiring SCOE level	OOL Condition detected by LLCS Wiring SCOE on Hardwired parameters and reaction executed by LLCS Wiring SCOE through Hardwired commands	30ms/40ms
Fast Reaction Loop at 1553 SCOE level	OOL Condition detected by 1553 SCOE on 1553 parameters and reaction executed by 1553 SCOE through 1553 commands	30ms/40ms
Reaction Loop 1553- LLCS Wiring SCOE	OOL Condition detected by 1553 SCOE on 1553 parameters and reaction executed by LLCS Wiring SCOE through Hardwired commands	65ms/85ms
Reaction Loop 1553-TPE- LLCS Wiring SCOE	OOL Condition detected by 1553 SCOE on 1553 parameters, event sent to TPE, which requests the LLCS Wiring SCOE to send Hardwired commands	120ms/140ms
Reaction Loop 1553-SCOS-TPE	OOL Condition detected by SCOS on 1553 parameters, event sent to TPE, which requests the LLCS Wiring SCOE to send Hardwired commands	150ms/170ms

4 CONCLUSIONS AND FURTHER WORK

The VEGA EGSE has been a successful experience. Robustness of design allowed supporting AIT activities already with preliminary releases of the EGSE. The choice at design time to bypass SCOS in the commanding chain was initially made assuming limitation on SCOS performances. Final benchmarks showed that passing through SCOS could have matched performance requirements. Nevertheless the implemented architecture opened the way to lighter architecture where the usage of SCOS could be considered excessive and where monitoring is not made over TM packets (e.g. payload EGSE, subsystems EGSE), but where it is required to apply ECSS-E-70-31 and ECSS-E-70-32. The scalability and the modularity of the presented architecture allow facing applications of every size and scope.

New frontiers for development are what we define “intelligent SCOEs” for small applications, where the minimal set of HLCS implementing the standards is embedded in the SCOE front end. This approach would extend enormously the granularity of the scope of our applications and allow homogenous management of the EGSE components throughout all the phases of integration, from subsystem to spacecraft.