Satellite Systems Functional Validation: from Specifications to Final Approval

Malik Khalfallah(1), Olivier Zanon(1), Claude Cazenave(1), Vincent Baulin(2)
(1)Airbus DS - Space Systems Engineering
31 rue des Cosmonautes, 31400 Toulouse France
malik.khalfallah@airbus.com, olivier.zanon@airbus.com, claude.cazenave@airbus.com

(2)Alten Paris
221 bis, boulevard Jean Jaurès 92514 Boulogne-Billancourt
vincent.baulin.external@airbus.com

ABSTRACT
This paper aims at outlining the process and the associated software that are used to conduct the validation tests for Eurostar Neo product line for telecommunication satellites as well as ASTROBUS product line for earth observation satellites. The presented software provides a matrix-like interface called Overall Test Matrix (OTM). The OTM allows the satellite systems engineers to develop test specifications while linking the different aspects of these specifications to the systems definition. Having this link avoids errors because the necessary properties of the systems that will be tested are extracted from the systems definition and not described in an informal text. In addition, the OTM differentiates between tests for generic satellite bus and tests for satellites that are derived from the generic tests. This leverages data reuse and permits formal computation of the differences between tests defined for the generic satellite bus and tests run on real satellites. Finally the OTM provides features to monitor the execution of tests and allows formal validation of results.

INTRODUCTION
We aim at developing a model driven approach for satellite validation. Our approach as depicted in Fig. 1 follows the general satellite architecture. This architecture consists of a generic satellite bus (platform) that defines the different satellite subsystems. This definition includes the properties of those subsystems as well as the TM/TC (Telemetries and Telecommands) data to cite a few. Once this platform is defined, it can be used as a basis for the forthcoming programs. An example of a satellite bus and the satellite based on it[1] is the Eurostar E3000 platform [1]. For certain programs and to respond to specific customer requirements, some customizations or changes can be performed at specific satellite level.

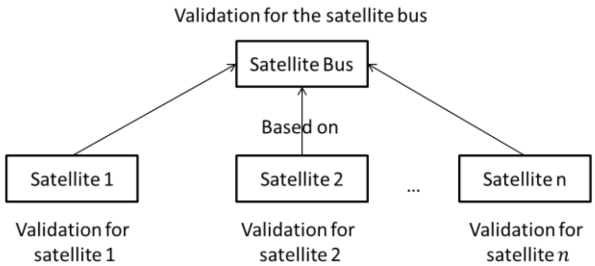

Fig. 1. Satellite bus architecture

By relying on this satellite architecture, our approach consists first in developing test specifications at the platform level. These specifications include necessary details such as how the test has to be conducted, test conditions such as the equipments necessary to perform the test, success criteria that can be used to evaluate the test result. An important rule for our approach is the notion of data reuse. Indeed, we aim at reusing as much as possible the data objects describing the platform. Following this rule allows us to maintain a link between the tests and the systems to be tested. Accordingly, if the systems properties are updated then the tests referencing those properties are updated as well without any involvement of satellite architects. In addition, as for the subsystems definitions that can be customized for a particular program, the test specifications can be customized as well to fit the program requirements.

Another added value of our approach is the capability to use it to monitor test executions at real time and also to validate their results. Indeed, in addition to linking tests with their systems, our approach links tests with their results. Accordingly there is an end-to-end link from the system specification to the test specification along with its results and their approval or rejection.

To achieve all these mentioned features and to make it easy for engineers to handle it, we have developed a matrix-based view called Overall Test Matrix (OTM) that engineers can build. This OTM provides an end-to-end process to:

[1] https://www.airbus.com/space/telecommunications-satellites/eurostar-series.html

1. Build a *generic OTM* that fits the satellite bus being developed
2. Define comprehensive test specifications and link them to the satellite bus structure in order to reuse all necessary subsystems properties
3. Associate test specifications with test procedures that implement them. Those procedures can be created for different test platforms (checkout systems) such as Open Center, EGSCC or CCS5.
4. Associate test specification with the AIT (Assembly, Integration and Testing) phases representing the conditions during which the tests will be run.
5. Generate program *specific OMT* that can be customized.
6. Generate program *dynamic OTM* that can be used to monitor the execution of tests and to sign them by approving or rejecting the results.

In the following sections, we detail this end-to-end process steps with illustration.

GENERIC OTM (STEP1)

The generic OTM aims at capturing test specifications for the satellite bus platform. An example of a generic OTM of a hypothetic platform called SAT_BUS is depicted in Fig. 2. The rows represent the test specifications of the SAT_BUS subsystems and the columns can be configured to manage the different details that have to be visible for test specifications. In this example the columns include the configuration of the satellite during the test as well as the AIT phases during which the test has to be performed.

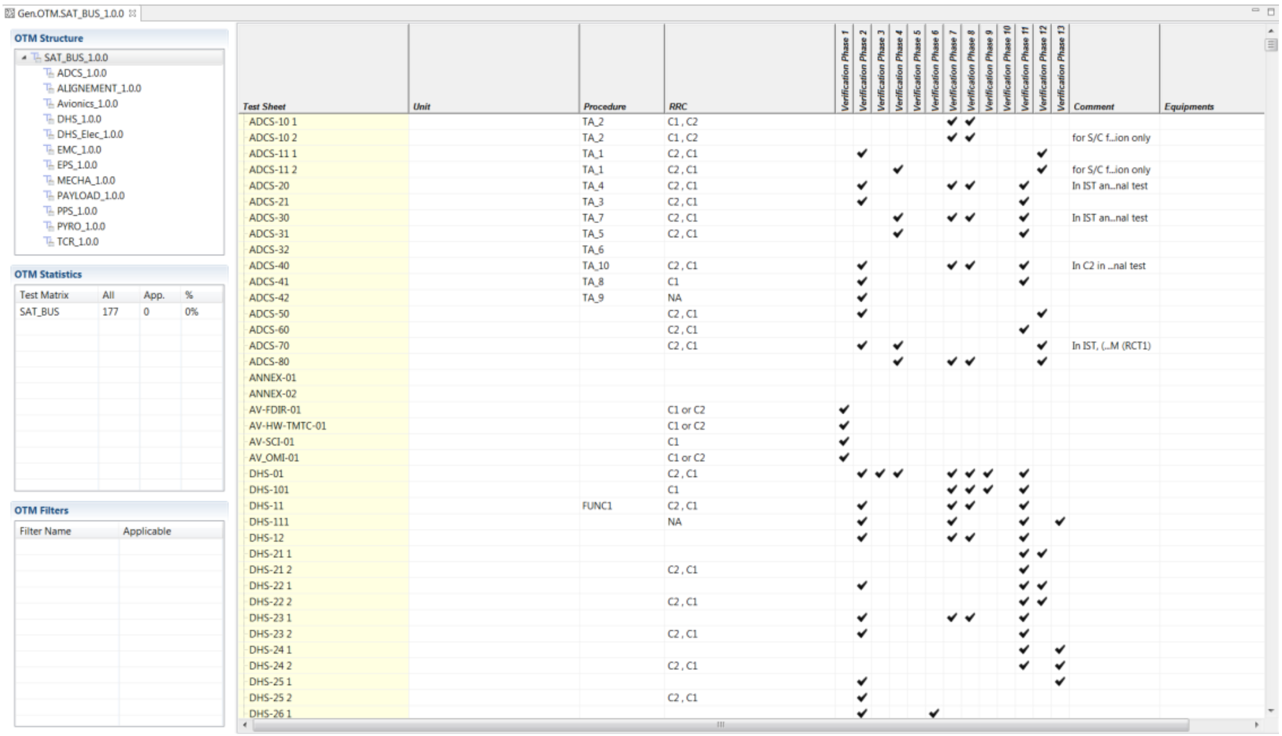

Fig. 2. An example of a generic OTM

TEST SPECIFICATION DEFINITION (STEP2)

The second step consists in defining the details of each test specification. These details concern mainly the properties of the subsystem that have to be captured and tested such as the telemetries and their expected values (success criteria). As depicted in Fig. 3 the front dialog allows engineers to select the success criteria for the ADCS-10 1 test specification. These success criteria objects can be used later on to generate parts of the procedure implementing that test specification. This avoids errors and lessens the maintenance burden for engineers. In addition to these objects being referenced, it is still possible to add informal descriptions of the test to give more details.

Engineers also have to select certain details in the OTM itself such as the configuration of the satellite that is required during the test (e.g. C1 to say all in nominal) as well as the AIT phases during which the test has to be run. The impact of associating objects with test specifications can be defined by the engineers themselves. For example engineers for a certain satellite bus would like that associating a test specification with a verification phase will by default associate all test specifications depending on this one to the same phase without any user involvement.

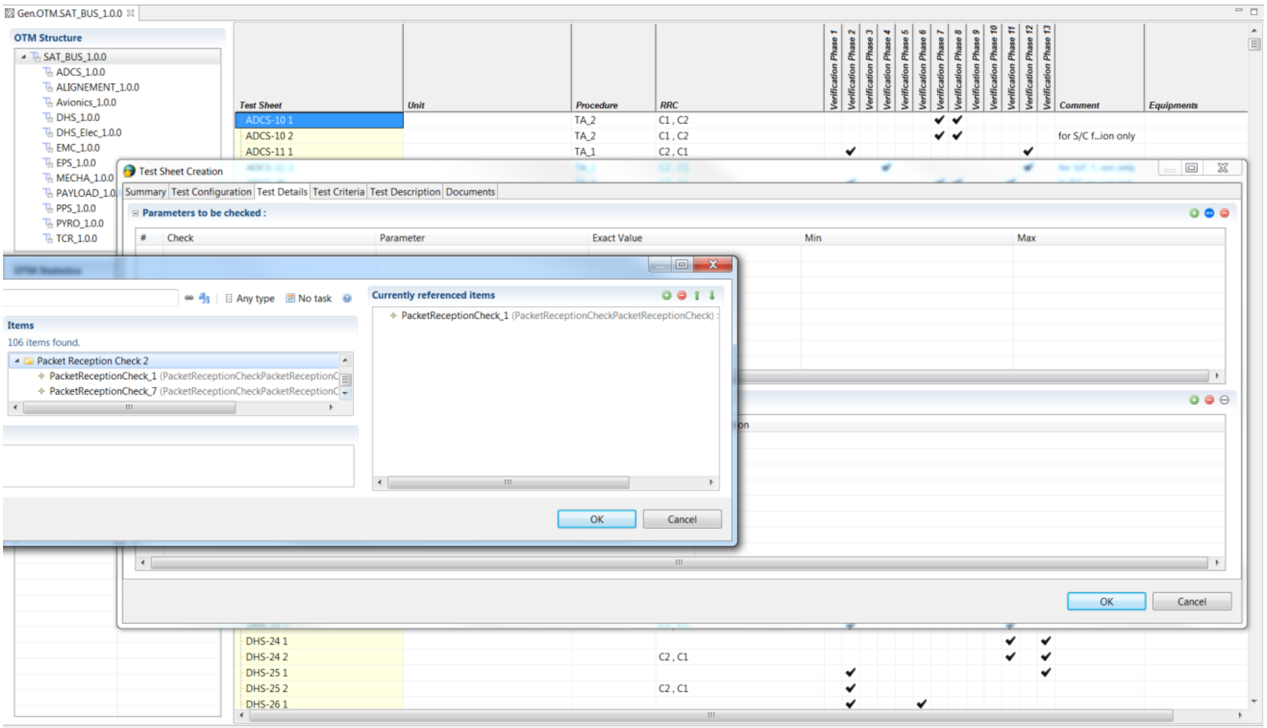

Fig. 3. Associating a test specification with systems properties

ASSOCIATION OF TEST SPECIFICATIONS WITH TEST PROCEDURES (STEP3)

Once engineers have defined their generic OTM, AIT engineers use that OTM details in order to define the test procedures (e.g. TA_1, TA_2 etc. written in a language such as Java) to implement each test specification. Then they associate the defined test procedures with their corresponding test specifications knowing that a single test procedure can implement multiple test specifications. The column Procedure in Fig. 2 shows the test procedures associated to some test specifications. Performing this association is necessary in order to allow the monitoring of the test later on.

SPECIFIC OTM (STEP 4)

Once the generic OTM has been defined for the SAT_BUS platform, the next step consists in generating the specific OTM based on this generic OTM for SAT_1 which is a program based on SAT_BUS. Engineers need to select the exact version of SAT_1 product structre for which they would like to generate an OTM as depicted in Fig. 4.

Generating a specific OTM for a given satellite program needs to join test specification data with the target satellite product data. Indeed, when defining test specification we said that they have to be associated with properties of the subsystem being tested. However if in the target satellite those properties have been updated and become different from the SAT_BUS properties, then the test specification cannot be kept in the OTM of the target satellite. As an example, suppose there is a test specification (DHS-01) in the generic OTM of SAT_BUS that aims at testing the equipment that ciphers TMs sent from the satellite to the ground station. If this equipment does not exist in SAT_1 satellite then there is no need to keep DHS-01 in the OTM of SAT_1.

Our specific OTM generation algorithm aims at handling cases like the mentioned one. In addition it aims at generating a specific OTM that is linked to its generic OTM. Keeping this (based-on) link is necessary for two purposes:

- Whenever a test is updated in the generic OTM, then the corresponding test in the specific OTM is updated as well
- Perform traceability and a highlight of all added or modified test details in the specific OTM. This could occur because customers might need to perform more or less tests on their satellite. Accordingly, it is necessary to capture the “delta” between the test specification in the specific OTM and the test specifications in the generic OTM.

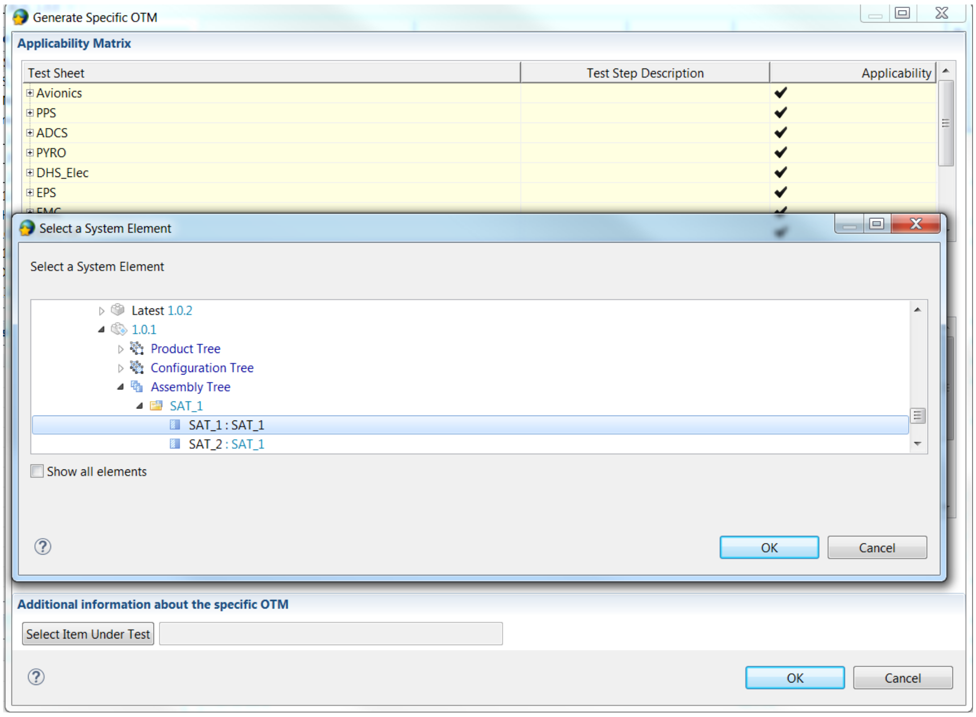

Fig. 4. Generating Specific OTM

Once the generation of the OTM is confirmed, then it is opened as depicted in Fig. 5. In this SAT_1 OTM we can see that it is similar to the generic OTM of Fig. 2. Nevrtheless, some colored cells exist. In fact these cells are colored because they have been updated in comparison to what exists in the generic OTM. In other terms, whenever a cell is updated in the specific OTM, then it will be highlited to show that there is a delta between the information in that cell and its corresponding cell in the generic OTM.

When a cell or other details of the test specifications have been updated in a specific OTM, then they will be decoupled from their corresponding entry in the generic OTM. Thus if the generic OTM cell value changes, then the correspoding specific OTM cell value will not be updated.

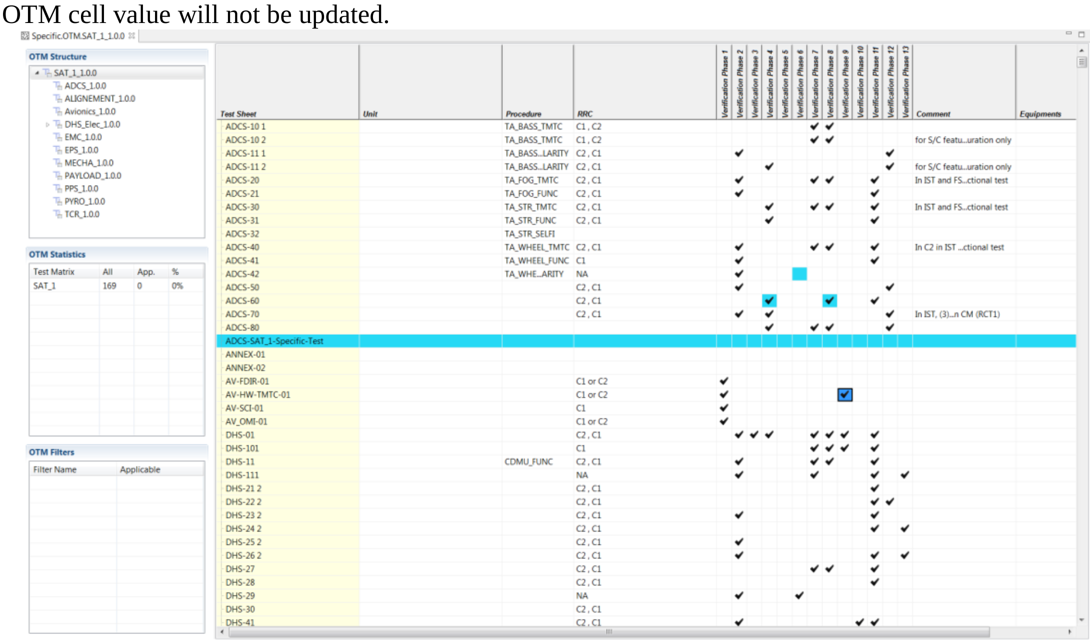

Fig. 5. Specific OTM of SAT_1 generated from the generic OTM of SAT_BUS with customizations highlited

To perform this customization, it is necessary for the engineers to explicitly select the field to be customized and then put the desired value. An example of customization is depicted in Fig. 6. The field “Parameters to be checked” has been customized. Once this customized is performed, it is possible to fill this table independently of the value of its corresponding entry in the generic OTM.

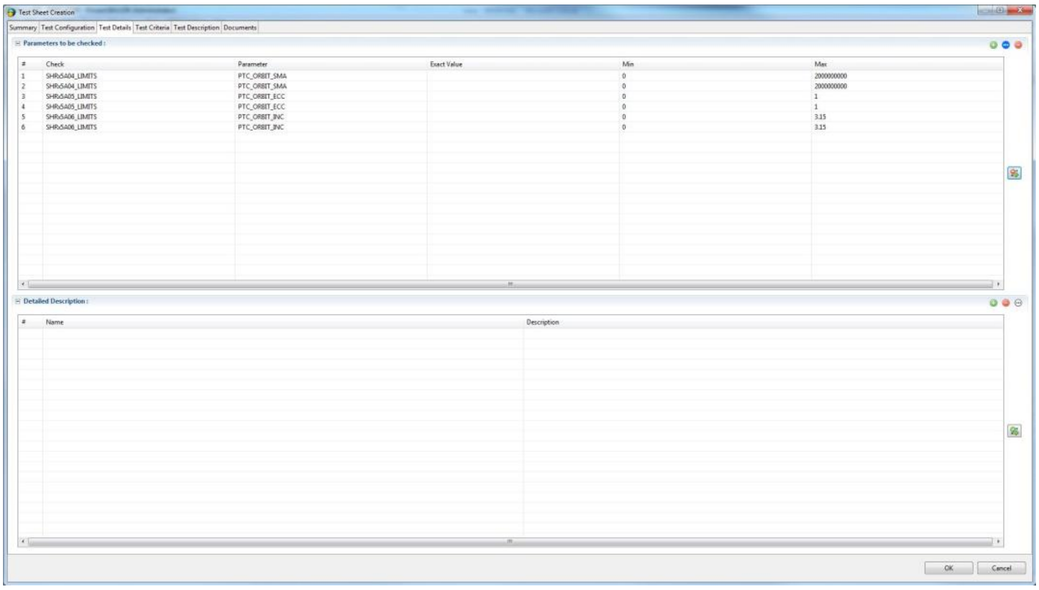

Fig. 6. Update of a test specification in SAT_1 OTM. An explicit cutomization button is visible on the right.

Once the specific OTM of SAT_1 has been generated and eventually cutstomized to meet the customer test requirements, it is possible now to use it to start monitoring the tests that will be perfomed in the AIT. When a specific OTM will be used to monitor tests execution, then we call it dynamic OTM.

DYNAMIC OTM (STEP 5)

The dynamic OTM is the specific OTM but with additional information. This information concerns:

- The statuses of the tests being run in the AIT
- The signatures of the tests whether approvals or rejections.

More specifically, the dynamic OTM will associate the test execution results as well as their approvals or rejections to the specifications of the test that were defined for a specific program.

Fig. 7 depicts a dynamic OTM. First in this figure, we can see that there is only one AIT verification phase. Indeed, this represents the currently active verification phase in the AIT. As AIT phases are sequential then there could be only one active verification phase at an instant$t$. Second, in this figure representing the dynamic OTM, we can see three additional columns (Level 1 signatures, Level 2 signatures, Level 3 signatures). Those columns represent the different levels of responsibility that will approve or reject a test.

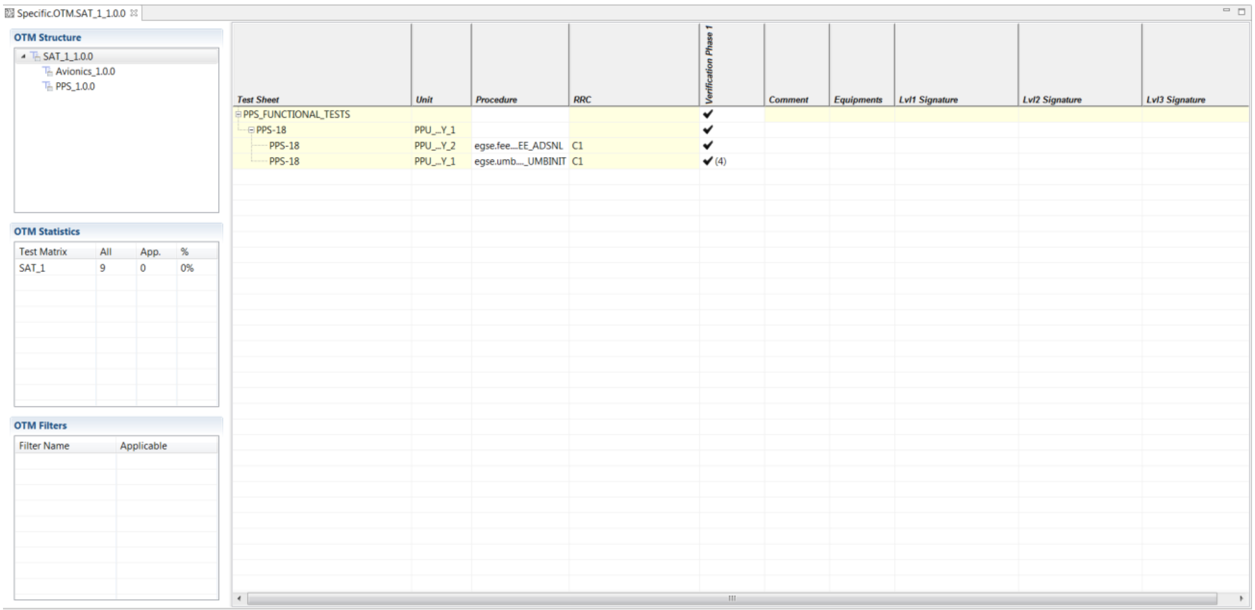

Fig. 7. Dynamic OTM

Third, in Fig. 7 we can see that in the last row there is a cell containing the number (4). This aims at indicating to engineers who are monitoring the execution of tests that the test in the last row has been executed 4 times and there are

4 results that need to be checked in order to sign the test. Such cells content is updated continuously and automatically by monitoring the test benches in the AIT.

If an engineer has the right role, then selecting the cell with the number 4 associated to it will open the dialog in Fig. 8. In this dialog we can see that all cells are in read-only mode except the cell Lvl1 Signature that the engineer can fill.

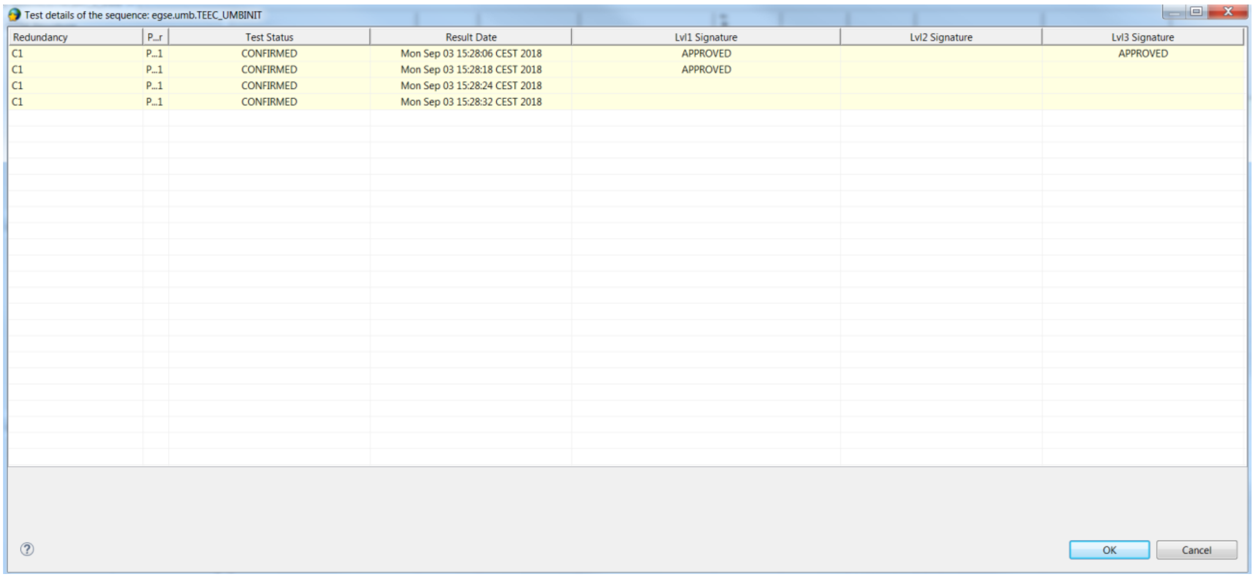

Fig. 8. Checking generated logs by a test

Nevertheless, before signing a test execution, the dynamic OTM offers the possibility to check the details of the generated logs related to the test. Fig. 9 depicts the list of those logs whose details can be checked. In addition, the OTM can invoke third party software (e.g. matlab, dynworks[6]) to perform finer analysis of the generated logs. It is necessary to have this capability because the OTM is not an analysis tool but can rely on other software or libraries to perform this work.
Once the engineers have checked the logs details and performed analysis on them if necessary, they can approve or reject a given log. This approval or rejection is accompanied by additional information. Indeed, a signature can be motivated by certain details and those details have to be kept with the signatures as depicted in Fig. 10. For that reason engineers have to associate a comment with their signatures as well as external files if necessary such as matlab report that are the results of deeper analysis. Those signature motivations can be viewed by other engineers who are in charge of the subsequent signature levels.

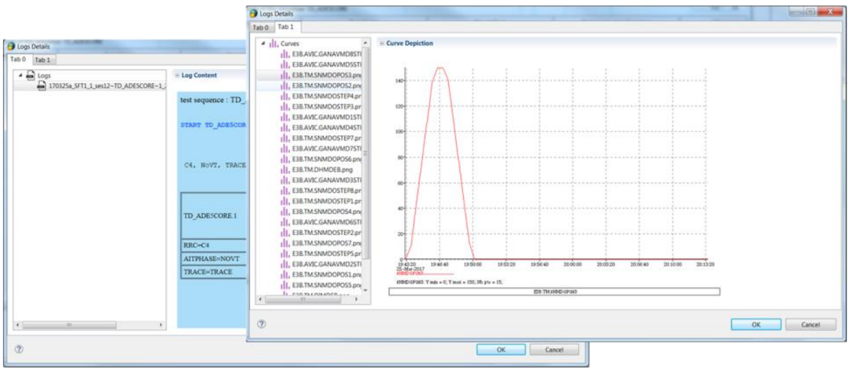

Fig. 9. Log details

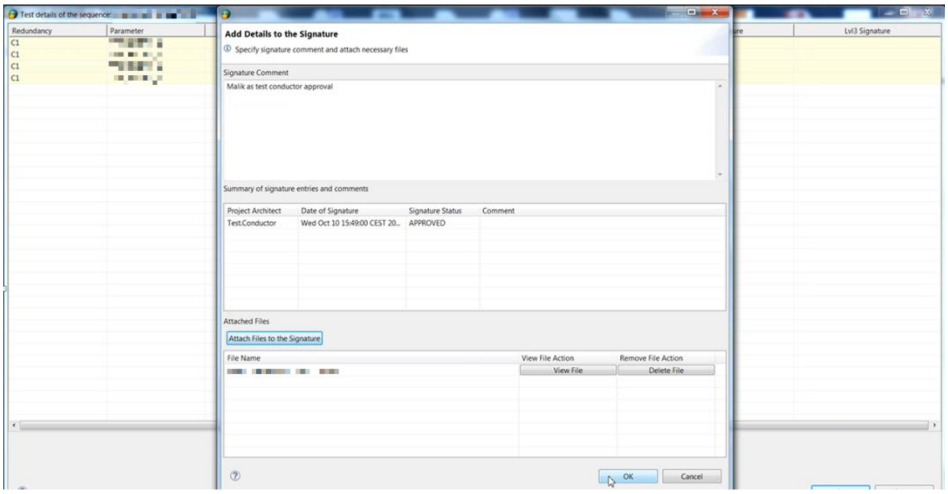

Fig. 10. Signature details

Once all signatures have been performed, the dynamic OTM is completed as depicted in Fig. 11. At that time, the OTM is tagged and put under configuration. In addition, it is possible to generate a report that can be used as a basis for the discussions with the customer regarding the tests that have been performed and the motivations behind any approvals or rejections.

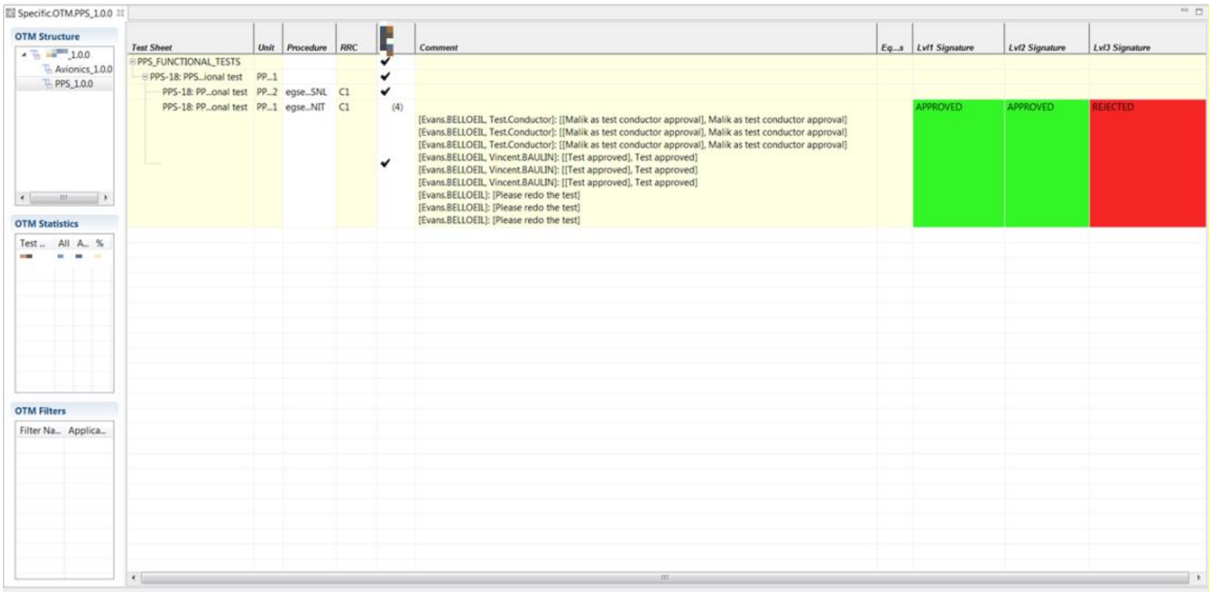

Fig. 11. Completed dynamic OTM

MULTIPLE PROGRAMS MONITORING

At this stage we have discussed the monitoring and the signatures of a single program testing using OTM. Nevertheless, it is possible for architects to open multiple OTMs of multiple programs to monitor and sign the tests that have been performed. This makes the OTM as a single entry point to validate multiple programs as depicted in Fig. 12.

SAT_BUS Generic OTM SAT_1 Specific OTM SAT_2 Specific OTM

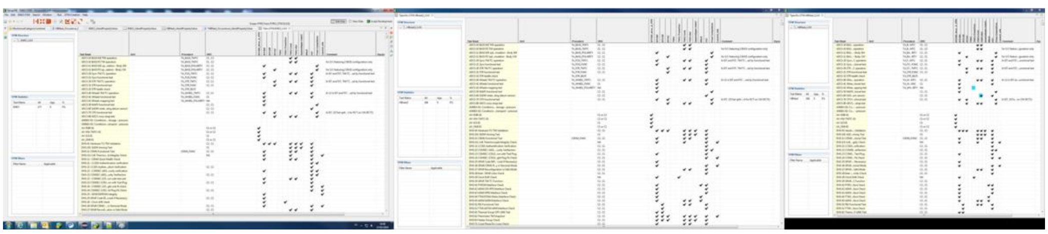

Fig. 12. Multiple OTMs to monitor the validation of multiple satellite programs in parallel

OTM ARCHITECTURE

Fig. 13 depicts the general architecture of the dynamic OTM. Multiple engineers can connect to monitor the execution of the validation tests on the satellite. Thanks to the monitoring approach put in place, it is possible to detect all necessary events and notify the engineers about them. Moreover, when engineers have the adequate access rights, then they can sign the tests that have been executed. These signatures are automatically propagated to all other connected engineers.

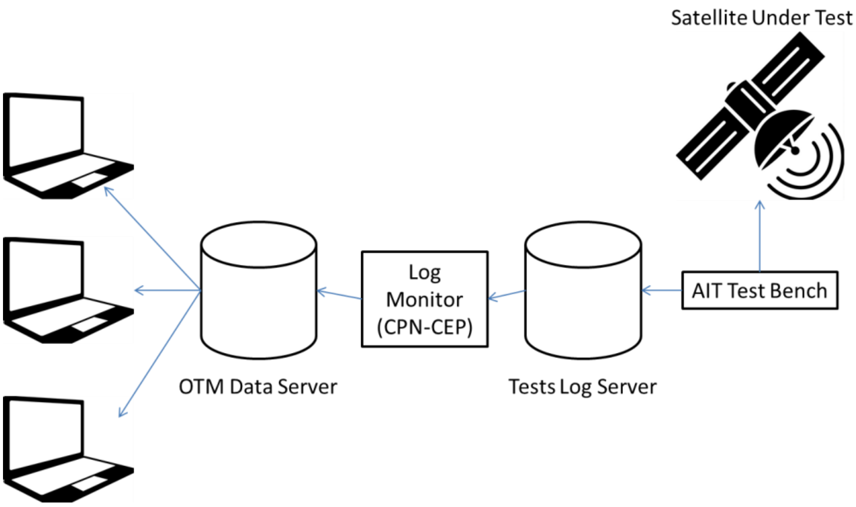

Fig. 13. OTM General Architecture

RELATED WORK

Verification is the proof that the space system meets the requirements and is in accord with the required project lifecycle [5]. Multiple standards and recommendations exist in the aerospace industry regarding the definition of processes of spacecrafts verification and validation. Among those we can cite NASA verification process [1] and ECSS Verification and Testing [2][3][4].

[1] proposed the Verification matrix (VMX) concept. They described the VMX as a "spreadsheet that contains the requirement's pre-verification data that is needed to perform the product requirement verification". Moreover, [2] proposed the Verification Control Document that is an excel table summarizing the details about the tests to be executed and their statuses.

The common aspect for these two approaches is that both rely on spreadsheets to specify and capture results of spacecraft tests. The drawbacks of such approaches are quite obvious. First they contain only unformal descriptions of the tests. Moreover there is no relationship with the definition of the product being tested.

CONCLUSION

In this paper, we have briefly presented the Overall Test Matrix (OTM) that is a model driven approach to develop test specifications for satellite systems. We have shown how the OTM can be used to support the end-to-end validation process for satellites. It starts by defining a generic OTM for the satellite bus. This OTM manages all tests that will be performed on satellites built using the referenced satellite bus. Then for each satellite program, a specific OTM is generated for that program. This generation takes into account the specificities of the target satellite. In addition all customizations performed on specific OTMs can be formally identified in order to compute the delta with the generic OTM. This might be useful for costing purposes for example. Once the specific OTM are generated, they are used to generate dynamic OTMs. The dynamic OTM for each program allows all engineers of the project to follow in real time the unfolding of tests in the AIT on the satellite. In addition if an engineer has the right role he/she can sign the results of the tests performed. At the end of the test cycle of the satellite, its dynamic OTM can be used as a proof showing that all tests have passed and delivered to the customer.

REFERENCES

[1] Alexander, M., Requirement Assurance: A Verification Process. NASA Langley Research Center, 2011.
[2] ECSS-E-10-02A - SPACE ENGINEERING: VERIFICATION
[3] ECSS-E-10-03A – SPACE ENGINEERING: TESTING
[4] ECSS-E-HB-10-02A – Verification guidelines
[5] Ley, W., Wittmann, K., Hallmann, W., Handbook of Space Technology, John Wiley & Sons 2011.
[6] Cavro, E. & Pont, C. Improvements in the DynaWorks® ISSPA identification module, Proceedings of the European Conference on Spacecraft Structures, Materials and Mechanical Testing 2005 (ESA SP-581). 10-12 May 2005, Noordwijk, The Netherlands.