AIMONS, V&V process update of GNC OBSW application

P. Dandré [(1)], T. Grossinho [(1)], F. Grisi [(1)], A. Beqiri [(1)], V. Pereira [(2)], I. Cantiello [(2)], D. Oddenino [(2)], B. Girouart [(2)]

[(1)] Thales Alenia Space – 5 allée des Gabians, 06150 Cannes (France)
name.surname@thalesaleniaspace.com
[(2)]ESA – Keplerlaan 1, 2201 AZ Noordwijk Netherlands
name.surname@esa.int,

ABSTRACT

Automatic code generation (ACG) has become a standard for GNC applicative software development in replacement of traditional manual specification. Model-based engineering with numerous tools for each step of the process has been developed to refine this newcomer ACG process. AIMONS study aimed at summarizing the lessons learnt from previous Thales Alenia space ACG V&V processes and at proposing any improvements required to the process. To reach this goal, the study experimented the process improvements in the PLATO use case.

The outcome is the proposal of a standardized Validation and Verification process and associated framework for GNC flight software developed with automatic code generation checked against the ECSS standards.

1 ABBREVIATIONS & ACRONYMS

ACG : Automatic code generation
ACP : Actuator Command Processing
ACT : ACTuator
AIMONS : ACG Improvement Methodology for ON-board Software
AOFT : AOcs FuncTion
APM : Antenna Pointing Mechanism
ATB : Avionics Test Bench
CDR : Critical Design Review
CTL : ConTroL
DKE : Dynamics, Kinematics and Environment
DYN : DYNamics
ECSS : European Cooperation for Space Standardization
FDIR : Failure, Detection and Isolation Recovery
FES : Functional Engineering Simulator
GDC : GuiDanCe
GNC : Guidance Navigation & Control
GTS : Galileo Transition Satellites
HiFi : High-Fidelity (simulator)
HIL : Hardware-In-the-Loop
ICD : Interface Control Document
IRD : Interface Requirements Document
MC : Monte Carlo
MIL : Model-In-the-Loop
MRS : Model Requirement Specification
NAV : NAVigation
OBSW : On-board SoftWare
PDR : Preliminary Design Review
PIL : Processor-In-the-Loop
PLATO : PLAnetary Transits and Oscillations of stars
PUS : Packet Utilization Standard
RB : Requirement Baseline
SADM : Solar Array Drive Mechanism
SAVOIR : Space AVionics Open Interface aRchitecture
SDB : Satellite Data Base
SDP : Sensor Data Processing
SEN : SENsor
SGS : SafeGuard Memory
SIL : Software-In-the-Loop
SRS : Software Requirement Specification
SSS : Software System Specification
SVF : Software Validation Facility
SW : SoftWare
SWOT : Strengths/Weaknesses/Opportunities/Threats
TRD : Technical Requirement Document
V&V : Verification and Validation

2 INTRODUCTION

Before entering into the specific considerations related to automatic code generation, the standard architecture of GNC simulators & OBSW application are presented highlighting the different perimeters on which code generation can be applied. The main architecture of a GNC simulator (FES) is depicted on the next Figure.

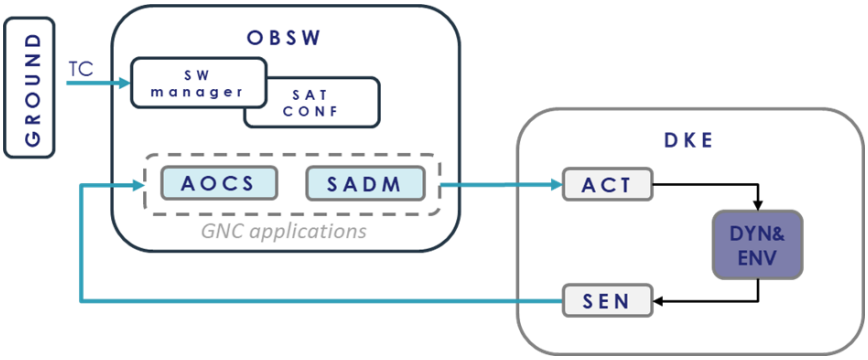

Figure 1. Standard GNC simulator architecture

The simulators are organized on two main modules: the on-board software (OBSW) and dynamic, kinematic and environment (DKE):

- OBSW module emulates the main software functions that interact with the GNC SW applications and integrates the GNC SW models, that will be generated directly from the simulator.
- DKE module simulates the main hardware elements (sensors and actuators) & the spacecraft dynamics and environment.

Depending on the programmatic aspects of each project, code generation could be applied at different levels:

- GNC elementary major functions also denoted AOFT,
- GNC modules including various elementary functions AOFT but without any mode logic,
- GNC SW application, including features such as the Satellite Data Base (SDB) and the definition of the different modes and its transition rules,
- Equipment models (sensors and actuators) and/or simulation of the spacecraft dynamics and environment (DKE).

Each perimeter imposes specific modelling rules. In particular, the flight code will be subjected to stringent rules of verification while the simulation of equipment and DKE blocks could be less restrictive in some cases.

The OBSW module acts as a wrapper to host the GNC application(s). It can also include some other functions that emulate key interface functionalities for the GNC modules such as telecommand dispatching, management of satellite configuration states, interface with internal memories (e.g. safeguard or mass memory) and any other function needed to feed the interfaces of GNC applications. These functions are generally not included in the code generation perimeter as they implement generic functionalities under responsibility of the software teams. At least, the tools potentially used for code generation considered for the latter features might not be the

standard ones used by GNC teams.

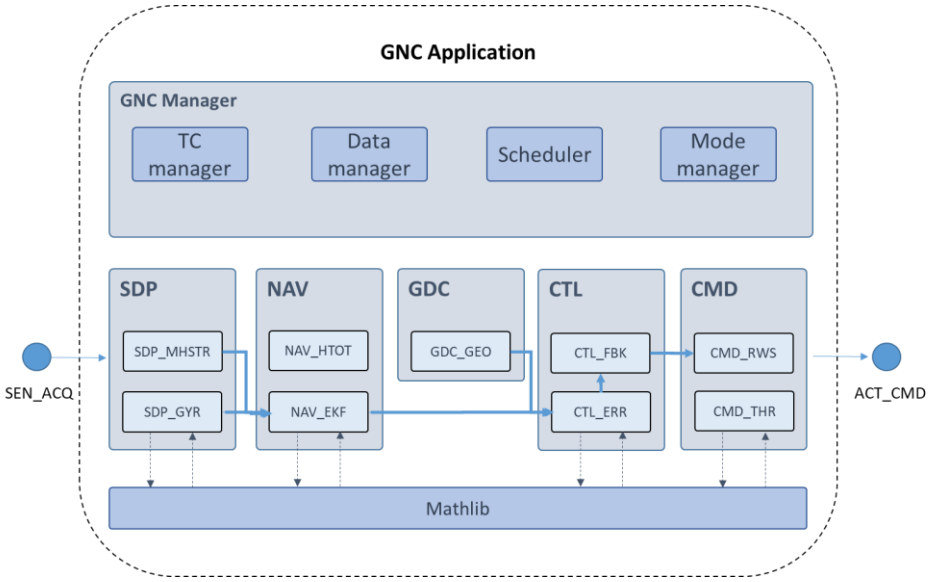

Figure 2. Standard architecture of GNC SW application

The GNC SW application can be decomposed in three different layers:

- First layer: manager functions, namely the functions needed to execute the telecommands, implement the mode logic (states initialization and reset) & organize the calls of the different elementary functions (scheduler).
- Second layer: GNC major functionality groups decomposed in sensor data processing (SDP), guidance (GDC), navigation (NAV), control (CTL) and actuator command processing (ACP).
- Third layer: GNC functions (AOFT). These functions may use additional layers of elementary mathematical functions (e.g. quaternion product).

The integration & interfaces of the autocode SW applications (AOCS & SADM) in the central software are synthetized in the following figure.

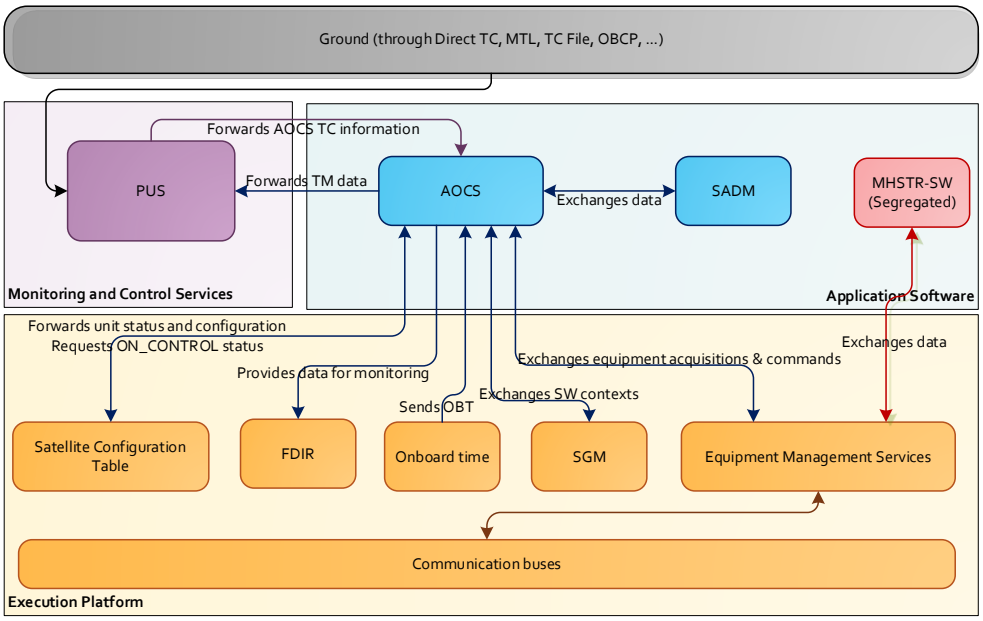

Figure 3. Integration & interfaces of GNC SW autocode application in central SW

3 THALES ALENIA SPACE LESSONS LEARNT

Lessons learnt are derived from various programs and products lines including among them Sentinel-3 mission (launch 2016), SBNEO product line (1st launch 2020), PLATO mission and PLATINO missions.

The autocode perimeter has been incremental along the different projects starting from a perimeter limited to core algorithms only and then moving step after step up to the complete GNC on-board software application.

The SWOT (Strengths/Weaknesses/Opportunities/Threats) of the autocode process is depicted in the next table.

Strengths	(initial) **Weaknesses**
▪ **Speed-up integration of design modifications** ▪ **Reduce non added-value activities** (late error detection, rewrite specifications with pseudo-code for available designs or algorithms) ▪ **Cost/schedule improvement** thanks to reduction of bugs due to manual specification (specification errors, misunderstanding, coding errors)	▪ Readability of the generated code ▪ Possible side effects on GNC performance verification based on High-Fidelity (HiFi) simulator: MIL (Model-In-the-Loop) campaign vs SIL (Software-In-the-Loop) campaign (computational performance, Low-layers equipment units data processing not modelled…) ▪ Risk of higher computational / Memory usage
Opportunities	**Threats**

▪ **Capitalization of coding guidelines**, merge all inherited processes into a unique process reusable for all upcoming developments ▪ Reuse of existing models developed during Phase B and reduce documentation workload ▪ **Anticipation of verification effort in MIL phase** ▪ Close partnership between GNC and SW engineers, improving skills of both teams ▪ GNC delivery product-oriented (confidentiality-IPR-wise)	▪ Dependency to external parties ▪ Dependency to tools under continuous evolution (with no warranty of maintenance of previous versions) ▪ Maintainability of the source code and configuration management at model level considering the possibility of several projects evolving in parallel ▪ Cost/Schedule impact of bugs generated by the code generation tool (non-certified tool, although no bug reported so far)

Table 1: Autocode process SWOT

The readability of the generated code might be a concern in case of willingness to modify the generated code without passing through the complete process. Even if not recommended, the situation could occur during critical operations (e.g. Launch and Early Operations Phase). Thanks to coding guidelines and precise configuration settings of the autocode tool, it is possible to highly improve the readability via imposing a code structuration and a sufficient level of comments.

The possible side effects of carrying out the GNC performances on a HiFi simulator with a different dynamics from the original simulator can be mitigated by applying the same autocode process to the environment models (actuators and sensors, dynamics). Alternately, the dynamics might be embedded into the GNC simulator that encompasses the GNC OBSW (On-Board SoftWare) models.

Initial risk of higher computational load and memory usage have not been encountered during Thales Alenia Space programs thanks to stringent coding guidelines and an efficient autocode tool that can be parametrized to optimize the memory stack for instance.

The next table synthetizes all Thales Alenia Space developments lessons learnt of the autocode process.

- Autocode process is highly recommended thanks to an efficient co-engineering approach: training sessions and clear definition of duties between subsystem and software teams are key to ensure a smooth transition to autocode process,
- A solid GNC algorithms unit test campaign (coverage part) performed in MIL will largely optimise the activities at OBSW-level, by capitalising from the MIL tests (additionally allowing to perform numerical proof of equivalence),
- Bugs are almost always discovered outside the autocode perimeter,
- Definition of internal guidelines for GNC developers is mandatory to ensure both an homogenous and fluent transition from model to source code,
- A large autocode perimeter is mandatory to fully benefit from the process. It includes OBSW GNC application and DKE,
- Optimal V&V process strongly depends on the autocode toolchains capabilities. Autocode tools cannot be decoupled from V&V process. Internal tools can be efficient especially for non-mature external tools,
- A constant technology survey of autocode tools and process is essential to fully benefit from latest enhancements (tools are still currently evolving at a significant rate),
- Verification at MIL level of GNC requirements can be very efficient to speed-up classical validation at ATB level, and proof of equivalence was achieved with relation to the SIL validation. Perimeter is to be carefully assessed to avoid duplication or leave gaps of activities between MIL, SIL, PIL and HIL validation stages,
- The required software documentation data packages create duplicated activities with relation to the autocode process and could be optimised.

4 PROPOSED V&V PROCESS & FRAMEWORK

Mathworks toolboxes related to the autocode process and subsequent validation have been assessed in comparison to internal Thales Alenia Space tool and relevancy of the toolbox with respect to the need in terms of qualification with ECSS standards. The following table collects the recommendations of the toolboxes.

Mathworks Tools	Tool recommendation	Major comment
Simulink Requirements	Reqtify	Both tools are possible. Reqtify tool is preferred for genericity with other OBSW applications and higher maturity at the time of the evaluation. Tool to be reassessed in the future notably for genericity and building blocks approach.
Simulink Coverage	Mathworks	Very efficient tool recommended for unit testing (coverage part) at MIL level.

Simulink Design Verifier	None	Current status is that formal verification (static code analysis) is proposed at SIL level to avoid rework between MIL and SIL
Simulink Test	Internal Thales Alenia Space tool (if available)	As most GNC requirements are verified on avionics test bench (via references runs comparison) and very few are strongly relying on MIL (GNC performances), use is currently not recommended. A simple internal tool is sufficient.
Simulink Checks	Mathworks	Efficient option to ensure conformance to the GNC guidelines (even if a higher maturity of the tool is needed).
Simulink Report Generator	Internal Thales Alenia Space tool (if available)	Use is currently not recommended. Given the current use, a simple internal tool can be sufficient to generate automatically the documentation.
Embedded Coder	Mathworks	Very efficient "Flight proven" tool (minor workarounds needed). Mandatory for automatic code generation.
Simulink Code Inspector	None	Never tested but more applicable to certification (model-to-code and code-to-model traceability). Current process could be satisfactory without.

Table 2: Autocode tool suite main lessons learnt & recommendations

The following figure presents the development process proposed by Thales Alenia Space as conclusion of AIMONS study. The major differences with respect to a standard process with manual specification are:

- Introduction of a Model Requirement Specification (MRS) as an input to the model,
- 2 parallel process branches: one for the autocode part (including notably the MRS, the models and the generated code) and one for the non autocode part (classical approach with RB/SRS),
- Modelling guidelines definition and verification step,
- Code coverage proposed to be anticipated since model level,
- Proof of equivalence between MIL and SIL for the GNC performance verification.

Note that the proposed process presents some updates with respect to the process presented in the ESA SAVOIR handbook [1].

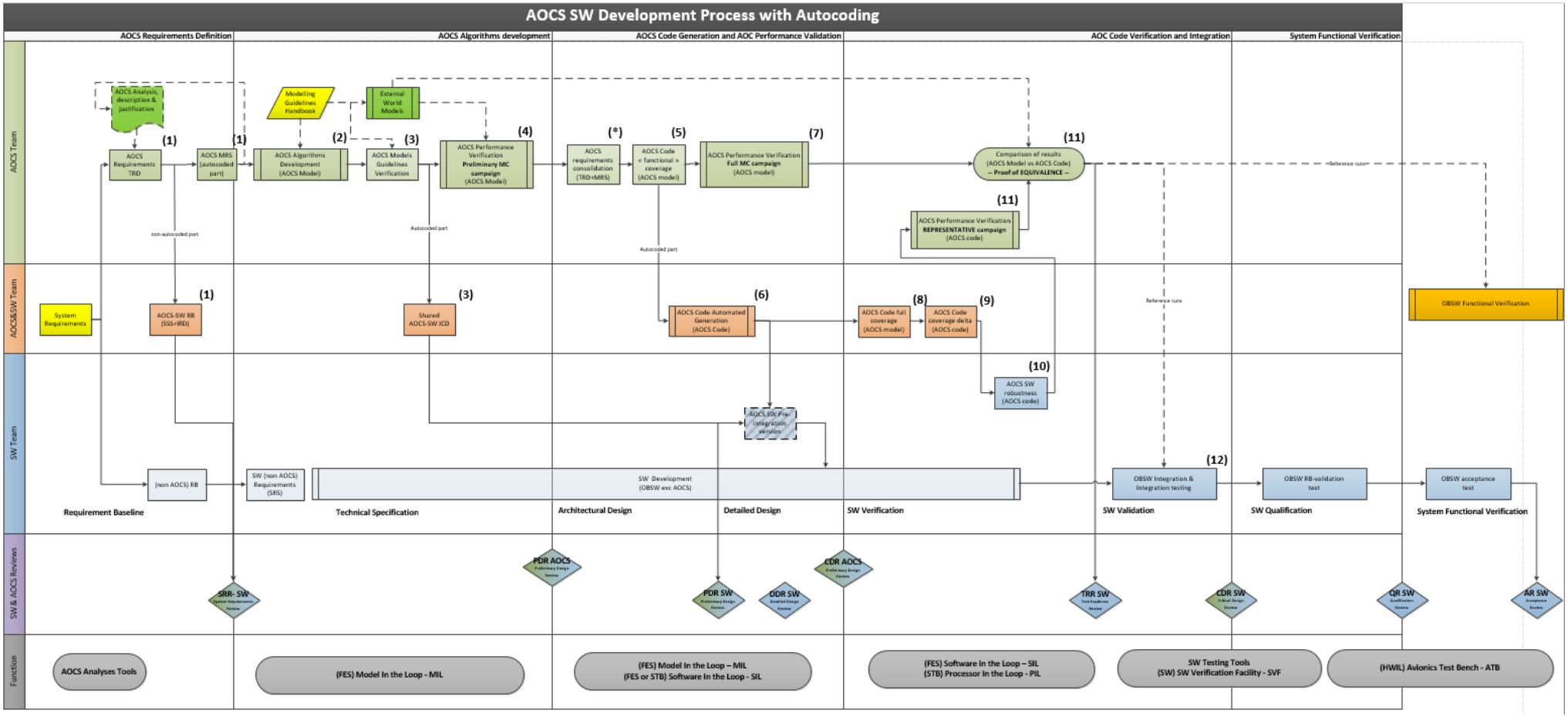

Figure 4. AIMONS proposed autocode process for AOCS SW application development

The different steps of the process and associated facility and framework are depicted hereunder.

Step	Name	Description	Facilities/Framework
(1)	GNC Requirements Definition	Derivation/apportionment of GNC requirements from System requirements and derivation to requirements baseline (SSS and IRD) for the non autocoded part & Model requirement specification (MRS) for autocoded part.	Doors or similar
(2)	GNC Algorithms Development (GNC Models)	Development and modelling of the GNC algorithms (following the modelling guidelines) Note: in parallel the models for external world (DKE, Sensors, Actuators, Environment) are developed and made available for the performance tests.	FES / MATLAB, SIMULINK
(3)	GNC Models Guidelines Verification (GNC Models)	The developed algorithms are subjected to Unit Testing where they are checked against modelling standard guidelines GNC/SW ICD are automatically generated by GNC and transferred to SW team.	FES / SIMULINK CHECK
(4)	GNC Performance verification Preliminary MC campaign (GNC Models)	GNC team runs preliminary performance verification campaign (normally Monte Carlo) to verify the compliance with GNC performance requirements	FES / MATLAB, SIMULINK
(5)	GNC Code Functional coverage (GNC Models)	The developed algorithms are subjected to Unit Testing where the "functional" coverage is checked. Functional coverage corresponds to the TRD branches (modes/submodes, guidance profiles, failure cases…)	FES / SIMULINK COVERAGE
(6)	GNC Code Generation (GNC Models to GNC SW)	Automatic Generation of the GNC SW Code	FES / MATLAB coder, SIMULINK coder & EMBEDDED coder
(7)	GNC Performance verification Full MC campaign (GNC Models)	GNC team runs full performance verification campaign (normally Monte Carlo) to verify the compliance with GNC performance requirements	FES / MATLAB, SIMULINK
(8)	GNC Code full coverage (GNC models)	The developed algorithms are subjected to Unit Testing where the full coverage is checked.	FES / SIMULINK COVERAGE
(9)	GNC Code coverage delta (GNC code)	The developed algorithms are subjected to Unit Testing on generated code to capture the corresponding SIL coverage.	FES / SIMULINK COVERAGE
(10)	GNC SW robustness (GNC code)	The developed algorithms are subjected to Unit Testing where the code robustness is checked as per ECSS E40C clause 5.5.3.2.	Internal tool

(11)	Proof of Equivalence test (GNC Model & GNC SW)	The Proof of Equivalence test is aimed at verifying that the generated code behaves as the model at numerical precision level. The test compares results from representative (sufficient coverage – metrics to be agreed within project – shall be granted together with sufficient excitation of functionalities) set of reference open loop tests cases run on the same test environment using GNC Models and GNC SW. Precision level is to be agreed within project. This corresponds to the validation of generated code with respect to Technical Specification represented by GNC Models. Note: in case the test is not successful (i.e. some differences cannot be explained) or as an alternative the GNC Generated Code shall be submitted to full GNC Performance campaign (e.g. Monte Carlo) on the SIL to verify the compliance with GNC performance requirements.	FES / MATLAB, SIMULINK, MATLAB coder, SIMULINK coder & EMBEDDED coder
(12)	OBSW Integration & Integration testing (GNC SW)	The generated and verified Code is delivered to the SW team to undergo its integration in the On Board SW (OBSW) for further qualification and acceptance testing before final delivery to system for functional verification (as per standard process).	SW Testing Environment (SVF, PIL, HIL)

5 EXPERIMENTATION ON USE CASES

The experimentation is mainly focused on the automatic verification of guidelines, on the code coverage analysis and on the usage of the respective toolboxes. The recommendations notably proposes the usage of Simulink check for automatic verification of the SAVOIR handbook guidelines and Thales Alenia Space internal guidelines. Simulink coverage might be also proposed pending the possibility to perform the code coverage at MIL level with efficient transition towards SIL level.

5.1 Automatic verification of guidelines

The use case chosen for the experimentation of the automatic verification is a subset of the PLATO service module: GNC and APM (Antenna Pointing Mechanism) on-board SW applications.

PLATO is a medium-class astronomical science mission belonging to ESAs Cosmic Vision Program, which is dedicated to the detection and characterisation of terrestrial exoplanets. PLATO orbit is the L2 Lagrange point of the Sun-Earth system. By performing long uninterrupted high precision observations of large samples of stars, the planetary transits can be detected and characterised by measuring and analysing the light curves of the stars. The operational lifetime of PLATO is 8.5 years (including lifetime extension). The spacecraft configuration consists of the following modules:

- Payload Module: contains the full set of instruments, an optical bench, supporting structures and the hardware thermal control,

- Service Module: supports the Payload Module by providing structural support, command and data management, attitude and orbit control, thermal control, communications, and power generation, conditioning and distribution.

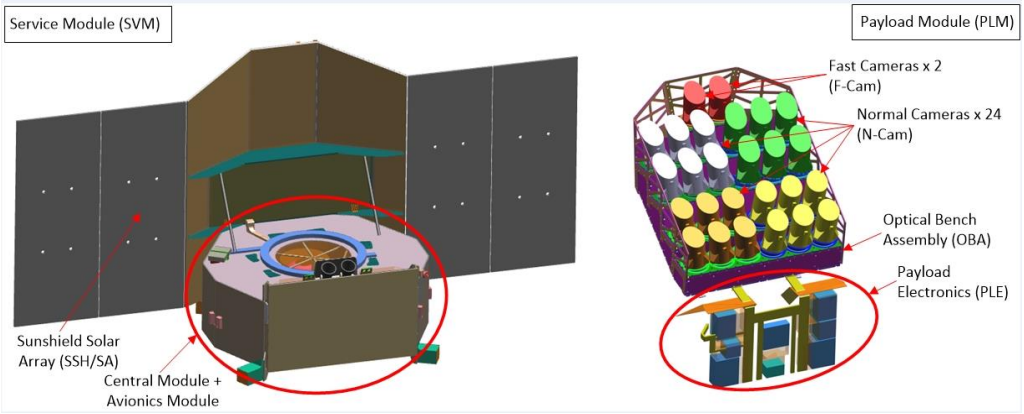

Figure 5. PLATO Modular Architecture

The perimeter of experimentation includes 59 requirements tested out of 98 requirements of the ESA SAVOIR handbook [1]. Thales Alenia Space internal guidelines was also experimented to enlarge the perimeter.

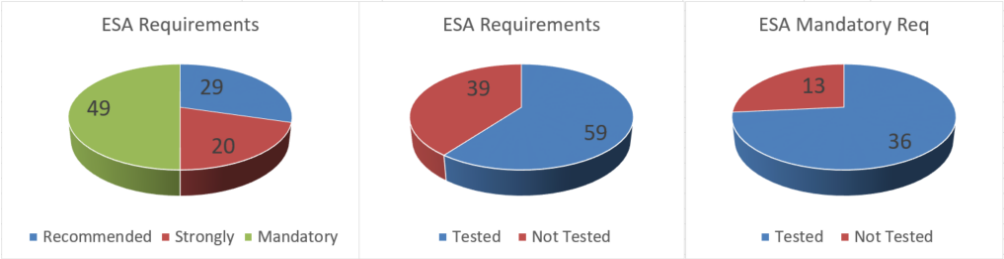

Figure 6. SAVOIR handbook experimentation perimeter

The experimentation includes both native checks provided by the toolbox and custom checks that can be implemented manually to customize the verification of a specific rule. 82 checks were implemented for the experimentation.

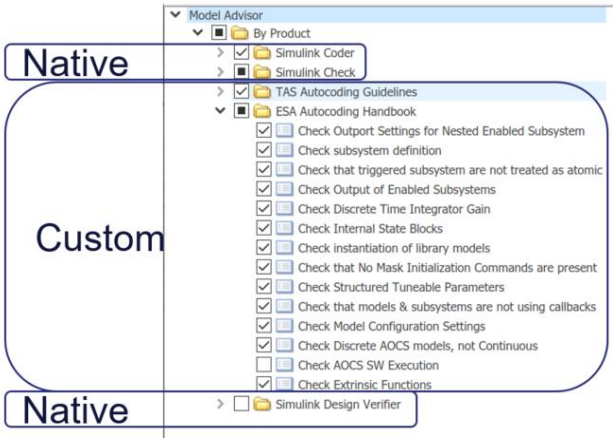

Figure 7. Simulink Check advisor configured for the experimentation

After an initial campaign and an enhancement of the toolbox configurations, the tools allowed to highlight several non-conformances to the guidelines. This non-conformances can be easily detected and corrected thanks to the tool report.

The experimentation is consequently considered fruitful. The tool is considered powerful and very easy to use; it offers the opportunity to run campaign via script. Some possible enhancements have been identified: exceptions treatments in a continuous integration process, stability of the toolbox with native checks differences between 2 releases.

5.2 Code coverage analysis

The experimentation of the code coverage analysis based on Simulink Coverage is done on 3 major GNC functions from PLATO and 3 major GNC functions from GTS. The main asset of the recent releases is that merge of coverage from numerous and different models is possible. It allows the combination of unit tests at major GNC function level and unit tests at complete GNC OBSW level.

Simulink Coverage performs model and code coverage analysis, implementing industrial standard metrics such as decision and condition.

The use of the toolbox in MIL allows the GNC engineer to anticipate the final coverage results of a test campaign (including both reference runs and unit tests) avoiding time consuming correction loops. The experimentation allows to analyse the delta between MIL and SIL code coverage campaigns and highlights possible limitations of the tool.

The report of one major GNC function is provided as an example.

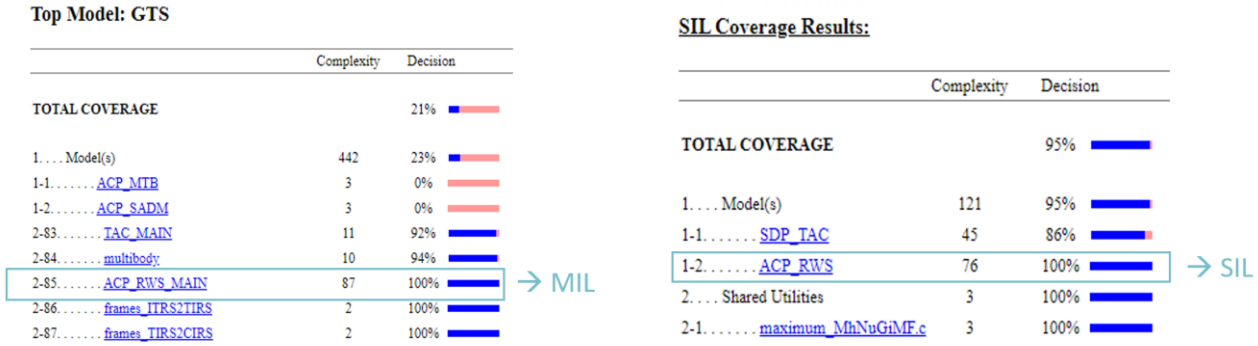

Figure 8. Simulink Coverage report example of one major GNC function

When done at the early stage of development, very few differences between the coverage in MIL and in SIL are observed. This is mainly due to architectural choices made during the autocode process design and tuning, such as:

- Coding and design rules proposed in the Thales Alenia Space guidelines, which enforce using only specific Matlab functions and reducing complexity of each individual algorithm,
- Additionally, it was seen that using the Simulink Check toolbox has proven to be a precious ally to verify the modelling and coding rules in the models, prior to generating the C code,
- Thales Alenia Space configuration parameters settings, which was tuned to reduce overhead in the code generation, as well as to mimic as much as possible the generated code to the manual Matlab code, still allowing code generation optimisations to be made (as long as they do not add overhead to the generated code).

The experimentation is consequently considered fruitful. The tool is considered powerful and very easy to use; it offers the opportunity to run code coverage campaign via script. Some possible enhancements have been identified: extra branches created during the autocode process that cannot be mastered by a specific configuration, non-obvious unexpected extra branches created at autocode generation that can be managed with an update of the guidelines as workaround.

6 FINDINGS & RECOMMENDATIONS

The proposed process and associated framework is fully compliant to ECSS standards with the deviations/clarifications stated hereafter:

- GNC PDR/CDR objectives strongly differ from SW PDR/CDR ones. It is proposed to maintain the two separated perimeters in dedicated reviews
- A detailed definition of the MRS document is to be proposed to clarify the purpose of the specification and delta with respect to the GNC Technical specification. Pseudo-code is not the expected level of specification but a sufficiently detailed requirements of the GNC function remains needed.
- Software unit testing: the code coverage is proposed to be fully verified at MIL level and in SIL (expected at least 95%, mainly due to delta derived from autocoding). Completion of MIL/SIL deltas to reach 100% SIL coverage can be reached via other methods (e.g. code inspection or analysis). The unit testing part described above, defined to achieve coverage purposes, does not replace the need to perform robustness, stress, out of range, and other type of unit testing on the SVF or other platforms.
- Proof of equivalence between MIL and SIL representative tests: the numerical precision level is composed of absolute/relative tolerances (notably for single/double float number) to be agreed with customer.
- Long term maintenance for flight SW: a maintenance plan of the GNC tool suites shall be indeed documented taking into consideration the tools support & maintenance time constraints.

7 ACKNOWLEDGMENT

AIMONS study has been co-funded by ESA research and innovation program under contract 4000133676/20/NL/CRS/kk. Thales Alenia Space has led the project.

8 REFERENCES

[1] ESA, *Guidelines for the Automatic Code Generation for AOCS/GNC Flight SW Handbook - SAVOIR Handbook*, 2022.