AUTOCODING WORKING GROUP

Automatic Code Generation for AOCS Flight SW

DAVIDE ODDENINO

12/11/2019

Special thanks for their valuable contribution to: Michele Pioli (trainee at ESA)
Francesco Grassini (TEC-SW) and the Extended Working Group

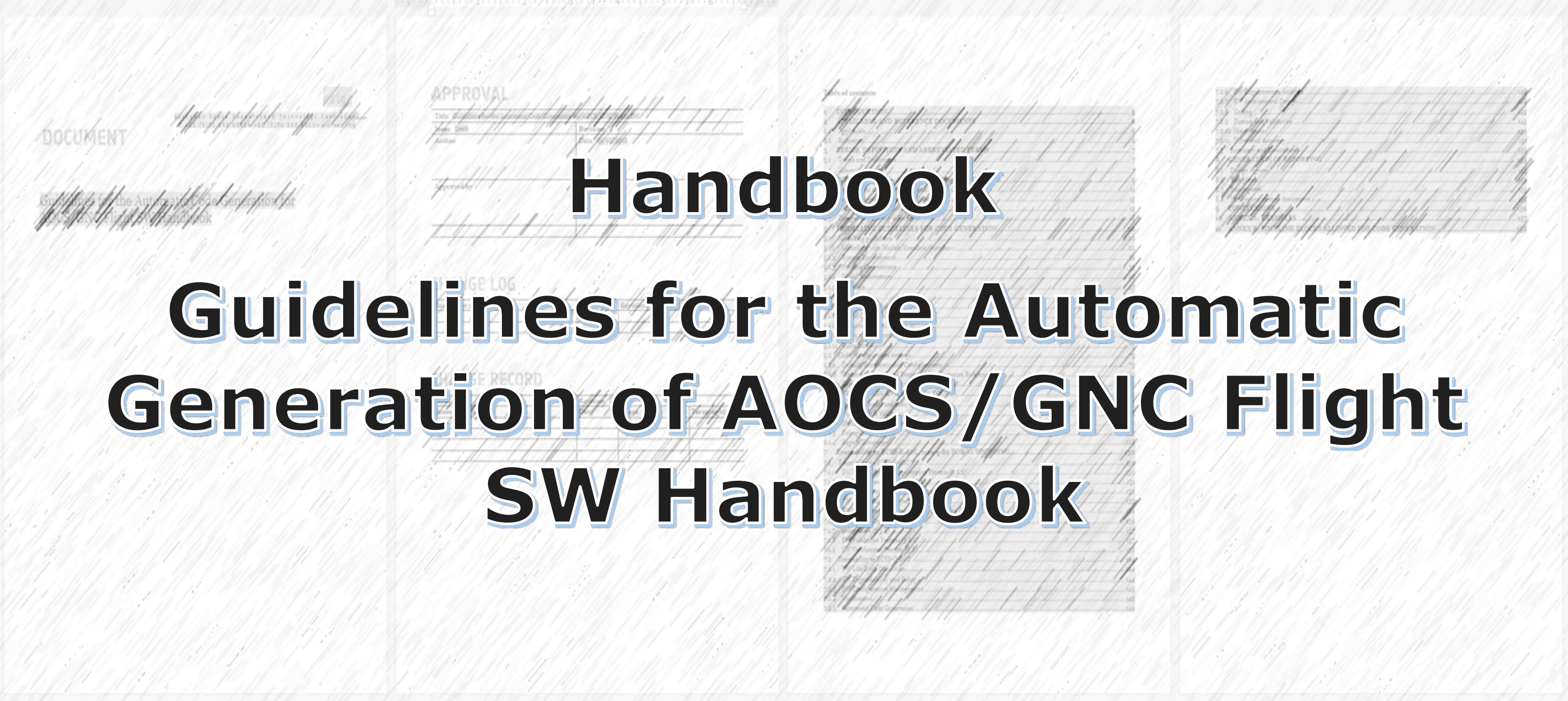
Outline of the presentation

- Background – Extended WG objectives
- Presentation of ESA Handbook
- Autocoding Process Definition
- Extended WG Major Comments overview
- Planning / Conclusions

SAVOIR AUTOCODE

space avionics open interface architecture

AUTOCODE Working Group



Handbook

Guidelines for the Automatic Generation of AOCS/GNC Flight SW Handbook

SAVOIR AUTOCODE

space avionics open interface architecture

AUTOCODE Extended Working Group

*The purpose of this Extended Working Group (EWG) is to **review the ESA Modeling guidelines for Autocoding Handbook** to be used as reference when creating models and generating flight code.*

The Handbook shall be used as reference with the objective of ensuring generated code is correct, reliable, readable, sharable/reuse-able and maintainable.

The intended use of the guidelines are the following ones:

- *Use in **support to projects** providing a harmonized ESA position across the Agency.*
- *Use in **R&D technology activities**.*
- *Promotion of the use of this type of methodology **across the phases of a development**.*
- *Contribution to the **assessment of the quality** of the final software product*

The scope of the Handbook includes

- *The **technology** (modelling guidelines, impact of the code generator, etc…)*
- *The **process** (GNC algorithm development process and application software process covering all the life cycle up to V&V)*

Modelling and Coding guidelines

General Modelling Guidelines

Approaches everything that has to do with the environment, in which the user models the system

ID	ESA-SY-001
Title	Consistent software environment
Priority	Mandatory
Description	During software development, it is recommended that a consistent software environment is used across the project. Software includes, but is not limited, to: - MATLAB - Simulink - C Compiler (for simulation) - C Compiler (for target hardware) Consistent software environment implies that the same version of the software is used across the full project. The version number applies to any patches or extensions to the software used by a group.
Rationale	If different versions are used there is no guarantee that the features will be compatible and the generated code is the same. This rule ensures the outcome is as expected.

Simulink

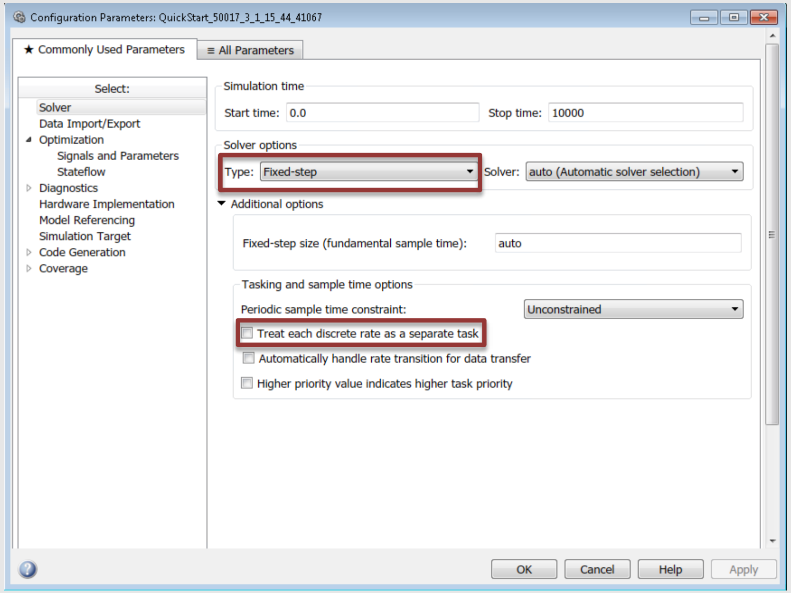
Rules regarding the Simulink blocks

ID	ESA-SL-001
Title	Blocks not recommended for C/C++ code production
Priority	Mandatory
Description	The model should not have any kind of blocks that are not suitable for code production. The list of such blocks can be used is in annex **Error! Reference source not found.**. Automatic Testing: mathworks.do178.PCGSupport mathworks.maab.jm_0001 mathworks.maab.hd_0001
Rationale	Using blocks compatible with code generation is essential for the process.

Generated Code Structure

These rules apply to the entire model

ID	ESA
Title	Parameter definition
Priority	Highly Recommended
Description	The parameters should be documented along with the class chosen for the parameter definition. It is recommended that parameters are defined either in the File Scope or in a general file containing all the OBSW parameters. Procedures and options on how to define parameter classes are demarcated in subsection **Error! Reference source not found.**.
Rationale	By defining beforehand how the parameters should be defined, it become predictable in which portion of the code the parameters will be declared and defined.



Configuration Parameters: QuickStart_50017_3_1_15_44_41067

Simulation time

Start time: 0.0 Stop time: 10000

Solver options

Type: Fixed-step Solver: auto (Automatic solver selection)

Additional options

Fixed-step size (fundamental sample time): auto

Tasking and sample time options

Periodic sample time constraint: Unconstrained

☐ Treat each discrete rate as a separate task

☐ Automatically handle rate transition for data transfer

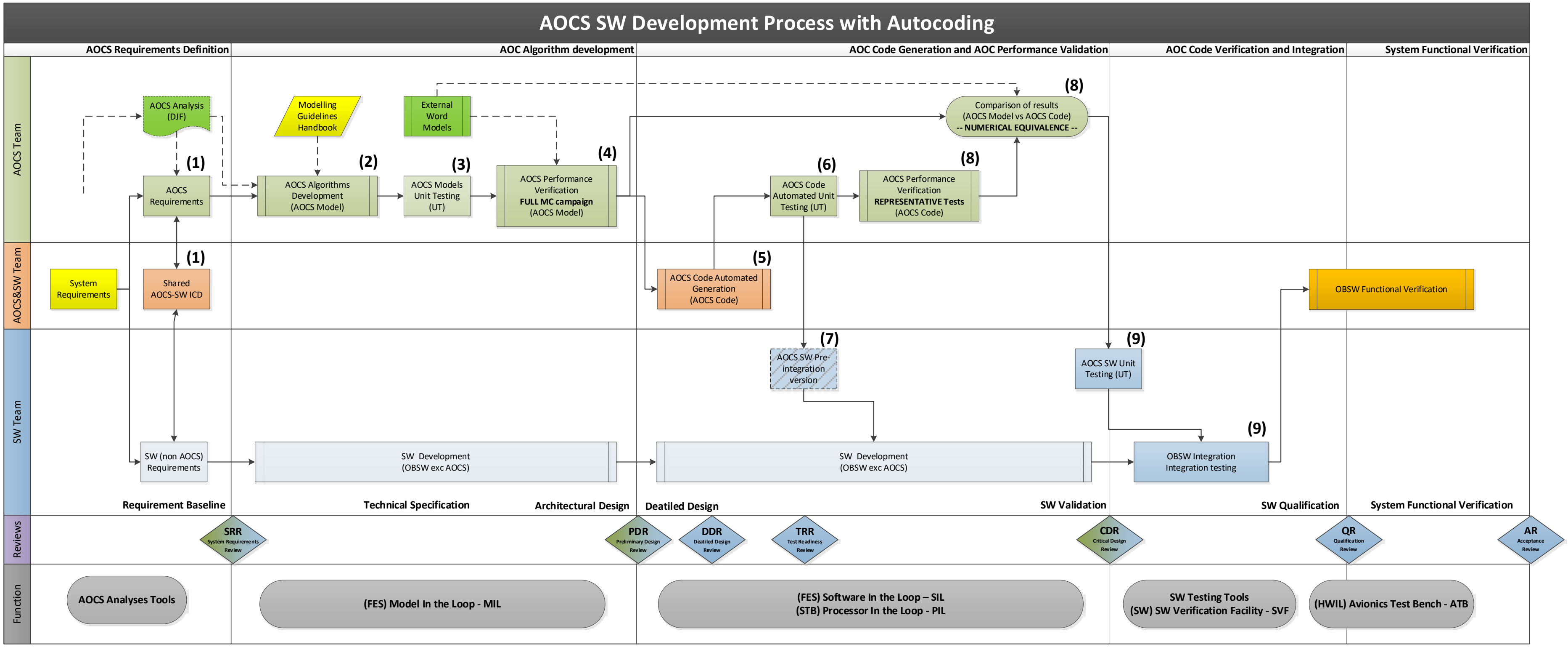
☐ Higher priority value indicates higher task priority

Parameter	Value	Description
Type:	Fixed-step	Required for code generation
Treat each discrete rate as a separate task	Unselected	Makes sure only one sample time (interruption in generated code) is generated.

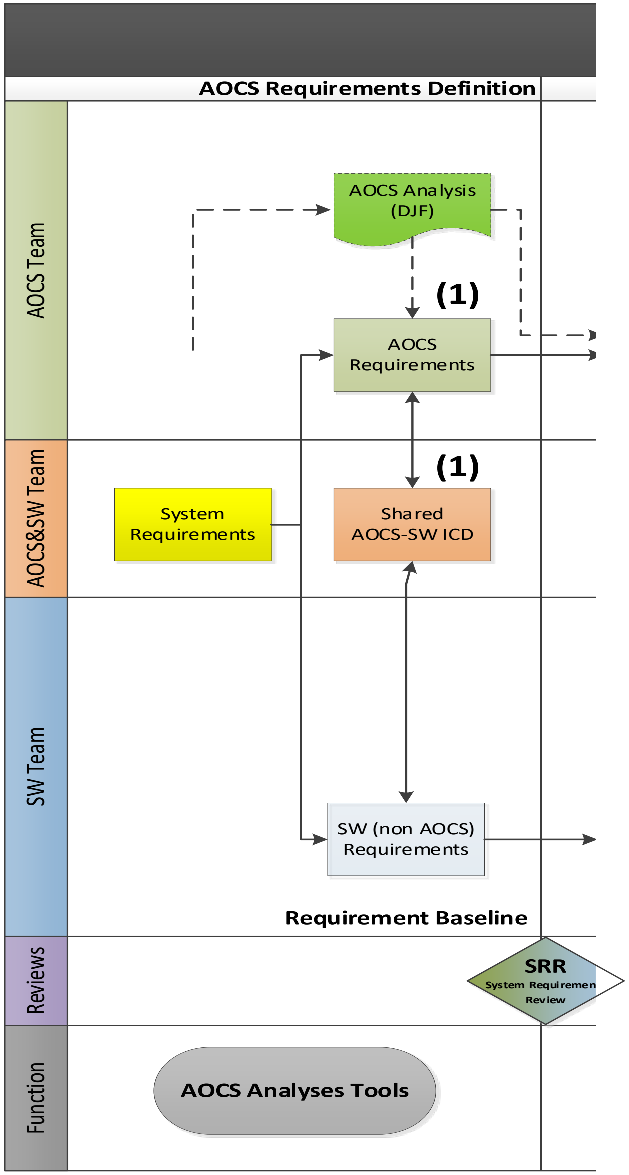
Traceability vs ECSS E-40 and ECSS Q-80

Clause	Description	Compliance
5.3.2.4	Automatic code generation a. The autocode input models shall be reviewed together with the rest of the software specification, architecture and design. NOTE The autocode input models are integral part of the software specification, architecture and design. EXPECTED OUTPUT: Autocode input model review [MGT, SDP; SRR, PDR]. b. In the case of coexisting autocoded and manually written parts, the software development plan shall include the definition of a clear interface definition and resource allocation (memory, CPU) at PDR. EXPECTED OUTPUT: Autocode interface definition and resource allocation [MGT, SDP; SRR, PDR]. c. The input model management, the code generation process and supporting tools shall be documented in the SDP. EXPECTED OUTPUT: Automatic code generation development process and tools [MGT, SDP; SRR, PDR]. d. The supplier shall define in the SDP the verification and validation strategy for automatic code generation as a result of the trade off between the qualification of the code generation toolchain and the end to end validation strategy of the software item, or any combination thereof, in relation with ECSS-Q-ST-80 clause 6.2.8. EXPECTED OUTPUT: Automatic code generation verification and validation strategy [MGT, SDP; SRR, PDR]. e. The configuration management of the automatic code generation related elements shall be defined in the SCMP. EXPECTED OUTPUT: Automatic code generation configuration management [MGT, SCMP; SRR, PDR].	a. Proposed in this HB: the model is part of the PDR, DDR reviewed by joint GNC/SW teams b. As proposed in this HB. In particular a SW/SW ICD between manual SW/GNC models and autocoded SW shall exist and be submitted to PDR c. This HB provided useful inputs for such Software Development Plan d. Qualification of the code generator is complex. Instead, this HB provide inputs for producing automated “qualifiable” code e. The approach to configuration management of model options, model toolchain shall be described in the SW Configuration Management Plan.

Automatic Code Generation approach of AOCS FSW

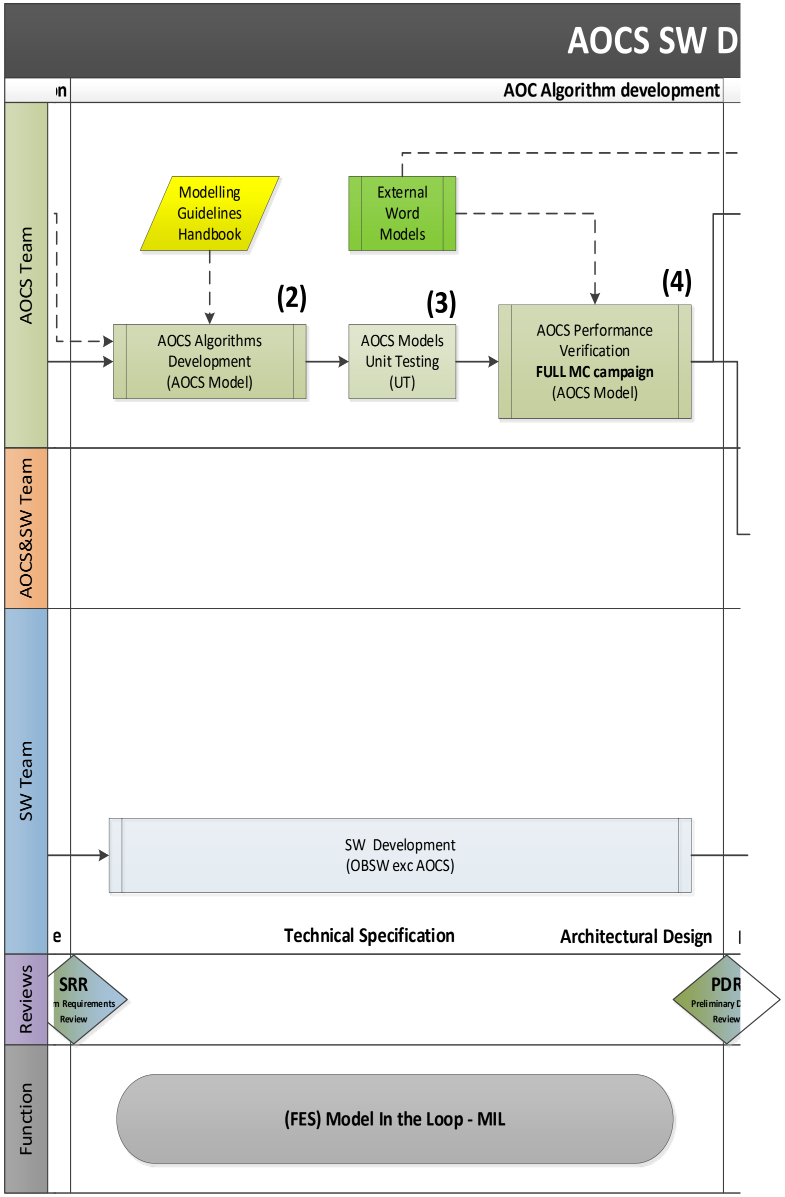

AOCS SW Development Process with Autocoding
AOCS Requirements Definition
AOC Algorithm development
AOC Code Generation and AOC Performance Validation
AOC Code Verification and Integration
System Functional Verification
AOCS Team
AOCS&SW Team
SW Team
Reviews
Function
AOCS Analysis (DJF)
Modelling Guidelines Handbook
External Word Models
(8)
Comparison of results (AOCS Model vs AOCS Code) -- NUMERICAL EQUIVALENCE --
(1)
AOCS Requirements
(2)
AOCS Algorithms Development (AOCS Model)
(3)
AOCS Models Unit Testing (UT)
(4)
AOCS Performance Verification FULL MC campaign (AOCS Model)
(6)
AOCS Code Automated Unit Testing (UT)
(8)
AOCS Performance Verification REPRESENTATIVE Tests (AOCS Code)
System Requirements
(1)
Shared AOCS-SW ICD
(5)
AOCS Code Automated Generation (AOCS Code)
OBSW Functional Verification
(7)
AOCS SW Pre-integration version
(9)
AOCS SW Unit Testing (UT)
(9)
OBSW Integration Integration testing
SW (non AOCS) Requirements
SW Development (OBSW exc AOCS)
SW Development (OBSW exc AOCS)
Requirement Baseline
Technical Specification
Architectural Design
Deatiled Design
SW Validation
SW Qualification
System Functional Verification
SRR System Requirements Review
PDR Preliminary Design Review
DDR Deatiled Design Review
TRR Test Readiness Review
CDR Critical Design Review
QR Qualification Review
AR Acceptance Review
AOCS Analyses Tools
(FES) Model In the Loop - MIL
(FES) Software In the Loop – SIL
(STB) Processor In the Loop - PIL
SW Testing Tools
(SW) SW Verification Facility - SVF
(HWIL) Avionics Test Bench - ATB

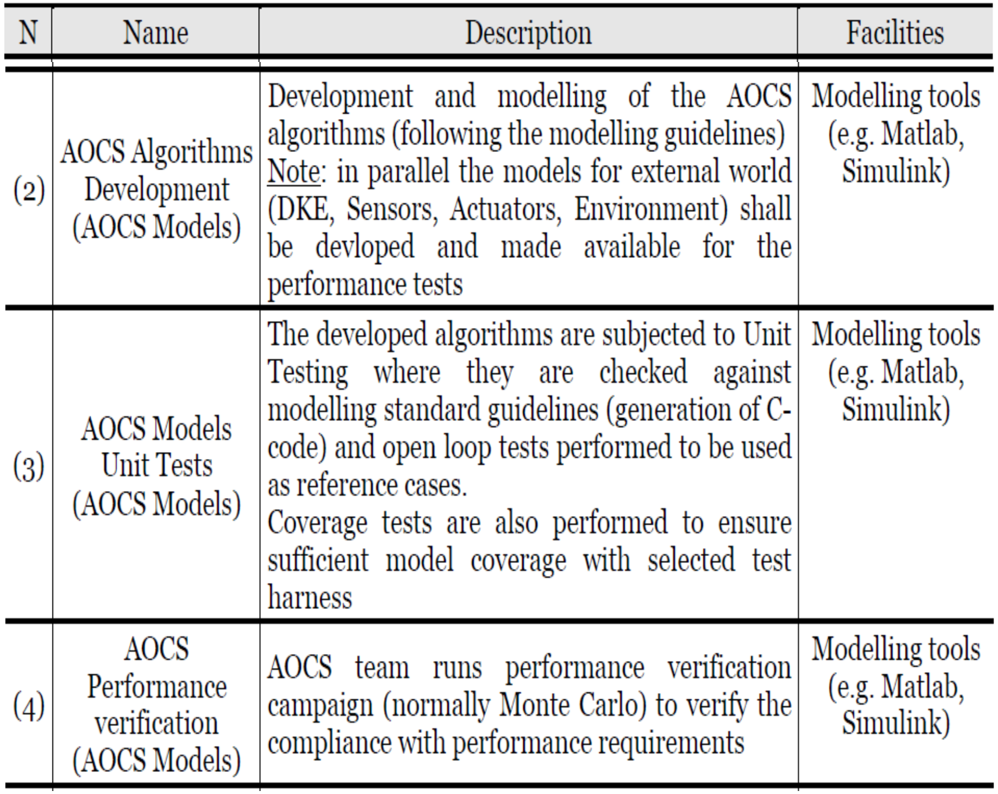
Automatic Code Generation approach of AOCS FSW



N	Name	Description	Facilities
(1)	AOCS Requirements Definition	Derivation/apportionment of AOCS requirements from System requirements. AOCS/SW ICD shall be developed together by AOCS and SW team	Analyses tools

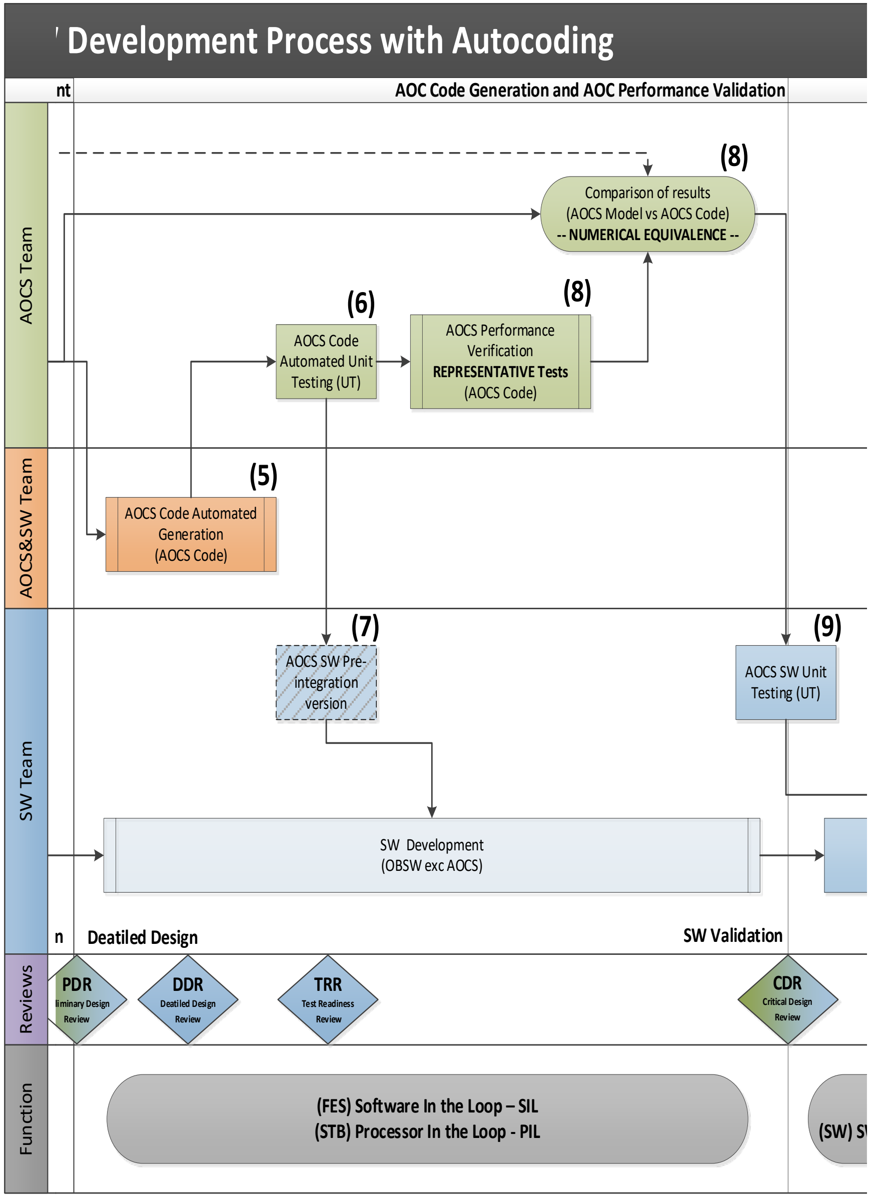
Automatic Code Generation approach of AOCS FSW


AOCS SW D
n
AOC Algorithm development
AOCS Team
Modelling Guidelines Handbook
External Word Models
(2)
(3)
(4)
AOCS Algorithms Development (AOCS Model)
AOCS Models Unit Testing (UT)
AOCS Performance Verification FULL MC campaign (AOCS Model)
AOCS&SW Team
SW Team
SW Development (OBSW exc AOCS)
e
Technical Specification
Architectural Design
Reviews
SRR
n Requirements Review
PDR
Preliminary D Review
Function
(FES) Model In the Loop - MIL



N	Name	Description	Facilities
(2)	AOCS Algorithms Development (AOCS Models)	Development and modelling of the AOCS algorithms (following the modelling guidelines) Note: in parallel the models for external world (DKE, Sensors, Actuators, Environment) shall be devloped and made available for the performance tests	Modelling tools (e.g. Matlab, Simulink)
(3)	AOCS Models Unit Tests (AOCS Models)	The developed algorithms are subjected to Unit Testing where they are checked against modelling standard guidelines (generation of C-code) and open loop tests performed to be used as reference cases. Coverage tests are also performed to ensure sufficient model coverage with selected test harness	Modelling tools (e.g. Matlab, Simulink)
(4)	AOCS Performance verification (AOCS Models)	AOCS team runs performance verification campaign (normally Monte Carlo) to verify the compliance with performance requirements	Modelling tools (e.g. Matlab, Simulink)

Automatic Code Generation approach of AOCS FSW



' Development Process with Autocoding

nt AOC Code Generation and AOC Performance Validation

AOCS Team

(8) Comparison of results (AOCS Model vs AOCS Code) -- NUMERICAL EQUIVALENCE --

(6) AOCS Code Automated Unit Testing (UT)

(8) AOCS Performance Verification REPRESENTATIVE Tests (AOCS Code)

AOCS&SW Team

(5) AOCS Code Automated Generation (AOCS Code)

SW Team

(7) AOCS SW Pre-integration version

(9) AOCS SW Unit Testing (UT)

SW Development (OBSW exc AOCS)

n Deatiled Design

SW Validation

Reviews

PDR liminary Design Review

DDR Deatiled Design Review

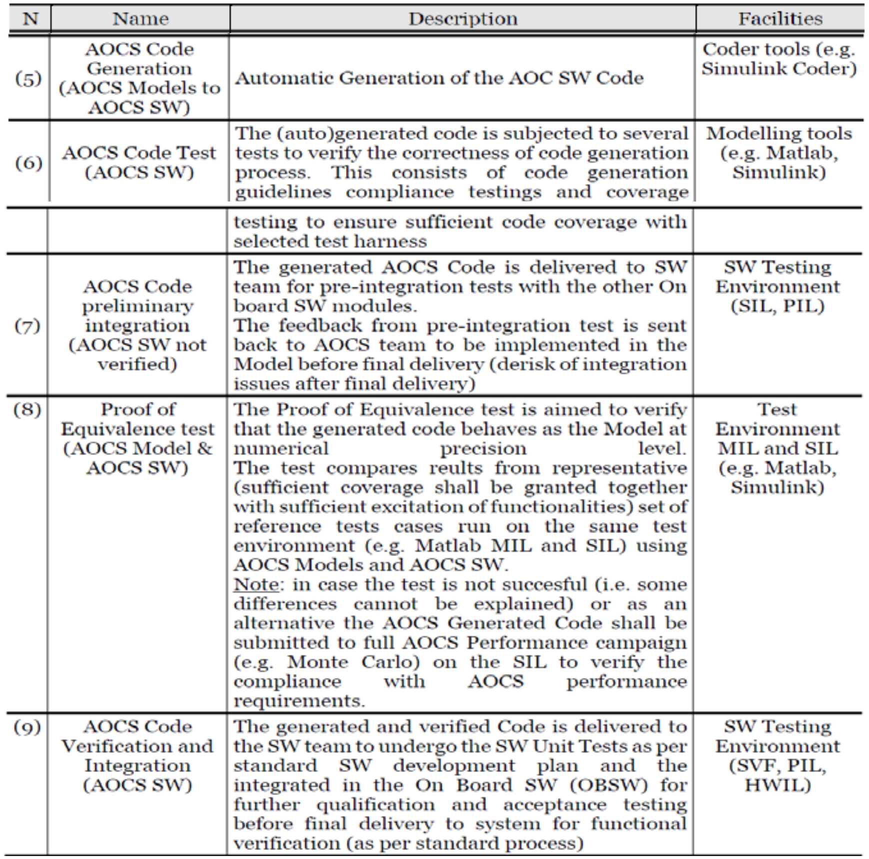
TRR Test Readiness Review

CDR Critical Design Review

Function

(FES) Software In the Loop – SIL
(STB) Processor In the Loop - PIL

(SW) S



N	Name	Description	Facilities
(5)	AOCS Code Generation (AOCS Models to AOCS SW)	Automatic Generation of the AOC SW Code	Coder tools (e.g. Simulink Coder)
(6)	AOCS Code Test (AOCS SW)	The (auto)generated code is subjected to several tests to verify the correctness of code generation process. This consists of code generation guidelines compliance testings and coverage testing to ensure sufficient code coverage with selected test harness	Modelling tools (e.g. Matlab, Simulink)
(7)	AOCS Code preliminary integration (AOCS SW not verified)	The generated AOCS Code is delivered to SW team for pre-integration tests with the other On board SW modules. The feedback from pre-integration test is sent back to AOCS team to be implemented in the Model before final delivery (derisk of integration issues after final delivery)	SW Testing Environment (SIL, PIL)
(8)	Proof of Equivalence test (AOCS Model & AOCS SW)	The Proof of Equivalence test is aimed to verify that the generated code behaves as the Model at numerical precision level. The test compares reults from representative (sufficient coverage shall be granted together with sufficient excitation of functionalities) set of reference tests cases run on the same test environment (e.g. Matlab MIL and SIL) using AOCS Models and AOCS SW. Note: in case the test is not succesful (i.e. some differences cannot be explained) or as an alternative the AOCS Generated Code shall be submitted to full AOCS Performance campaign (e.g. Monte Carlo) on the SIL to verify the compliance with AOCS performance requirements.	Test Environment MIL and SIL (e.g. Matlab, Simulink)
(9)	AOCS Code Verification and Integration (AOCS SW)	The generated and verified Code is delivered to the SW team to undergo the SW Unit Tests as per standard SW development plan and the integrated 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, HWIL)

SAVOIR AUTOCODE

space avionics open interface architecture

AUTOCODE Extended Working Group

Major comments overview

Collection of comments and observations to the HB draft

TAS, ADS, OHB, GMV, CNES, DLR

Mathworks has been also involved focused on details about the toolboxes

Extended Working Group – comments (1/5)

ESA-SE-050: Library Blocks Defined as Atomic
All the library blocks (e.g. mathematical, guidance, navigation, control models, etc.) shall be set as atomic subsystem with explicitly specified interfaces for data type and dimension.

Model reference is the only way to ensure deterministic, reusable C code. Reusable functions shall be defined as **reference model** to ensure that same code is generated and strong definition of I/O ports to ensure the consistency.

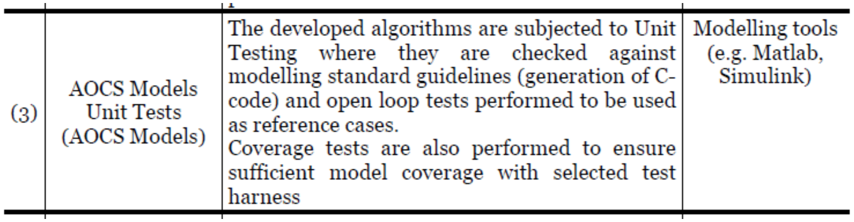
ESA-SE-010: Consistent Software Environment
During software development, it is recommended that a consistent software environment is used across the project. Consistent software environment implies that the same version of the software is used across the full project

It will be clarified that the freezing of the develooment environment release (e.g. Matlab version) shall be in place at the proof of equivalence step, just before the code generation process.

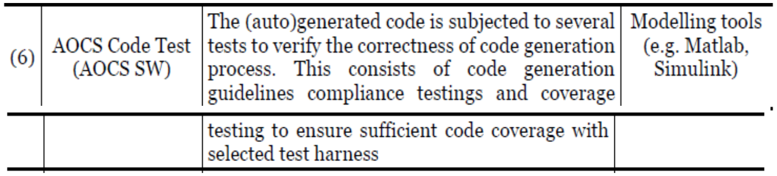
Extended Working Group – comments (2/5)

Autocoding Process: Unit Testing on Model, Code and generated SW

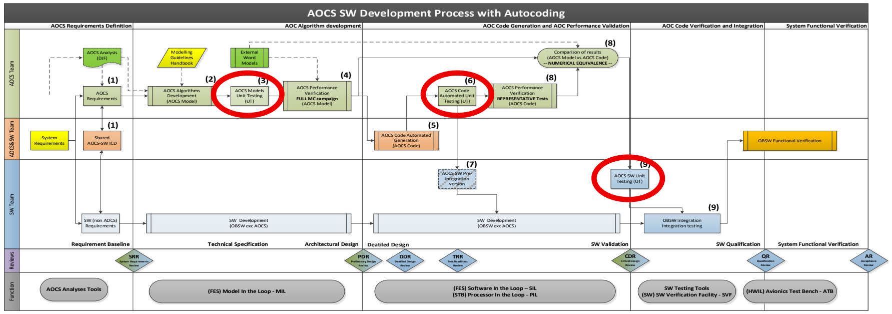
Why Unit testing at different levels are necessary?



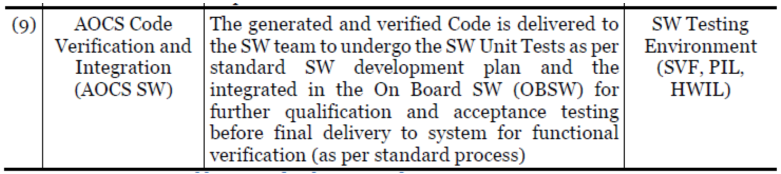
(3)	AOCS Models Unit Tests (AOCS Models)	The developed algorithms are subjected to Unit Testing where they are checked against modelling standard guidelines (generation of C-code) and open loop tests performed to be used as reference cases. Coverage tests are also performed to ensure sufficient model coverage with selected test harness	Modelling tools (e.g. Matlab, Simulink)



(6)	AOCS Code Test (AOCS SW)	The (auto)generated code is subjected to several tests to verify the correctness of code generation process. This consists of code generation guidelines compliance testings and coverage testing to ensure sufficient code coverage with selected test harness	Modelling tools (e.g. Matlab, Simulink)



AOCS SW Development Process with Autocoding
AOCS Requirements Definition
AOC Algorithm development
AOC Code Generation and AOC Performance Validation
AOC Code Verification and Integration
System Functional Verification
AOCS Team
AOCS&SW Team
SW Team
Reviews
Function
AOCS Analysis (DJF)
Modelling Guidelines Handbook
External Word Models
(1)
AOCS Requirements
(2)
AOCS Algorithms Development (AOCS Model)
(3)
AOCS Models Unit Testing (UT)
(4)
AOCS Performance Verification FULL MC campaign (AOCS Model)
(6)
AOCS Code Automated Unit Testing (UT)
(8)
AOCS Performance Verification REPRESENTATIVE Tests (AOCS Code)
(8)
Comparison of results (AOCS Model vs AOCS Code) -- NUMERICAL EQUIVALENCE --
System Requirements
(1)
Shared AOCS-SW ICD
(5)
AOCS Code Automated Generation (AOCS Code)
OBSW Functional Verification
(7)
AOCS SW Pre-integration version
(9)
AOCS SW Unit Testing (UT)
(9)
OBSW Integration Integration testing
SW (non AOCS) Requirements
SW Development (OBSW exc AOCS)
SW Development (OBSW exc AOCS)
Requirement Baseline
Technical Specification
Architectural Design
Deatiled Design
SW Validation
SW Qualification
System Functional Verification
SRR System Requirements Review
PDR Preliminary Design Review
DDR Deatiled Design Review
TRR Test Readiness Review
CDR Critical Design Review
QR Qualification Review
AR Acceptance Review
AOCS Analyses Tools
(FES) Model In the Loop - MIL
(FES) Software In the Loop – SIL
(STB) Processor In the Loop - PIL
SW Testing Tools
(SW) SW Verification Facility - SVF
(HWIL) Avionics Test Bench - ATB



(9)	AOCS Code Verification and Integration (AOCS SW)	The generated and verified Code is delivered to the SW team to undergo the SW Unit Tests as per standard SW development plan and the integrated 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, HWIL)

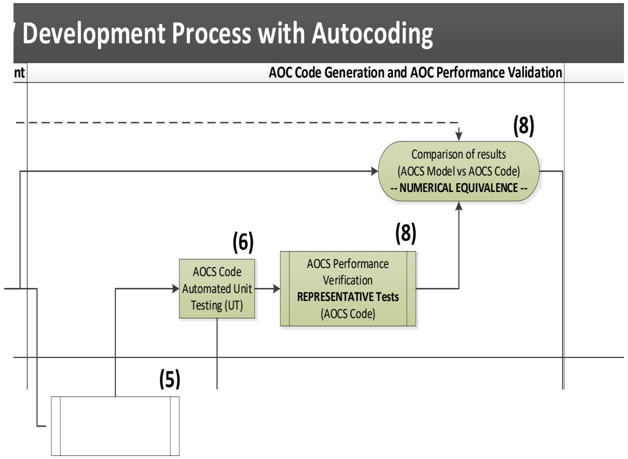
The Model Unit Tests are different in scope and execution from SW Unit Tests and therefore they cannot be skipped.

SW unit tests are considered complementary tests aiming to achieve compliance with ECSS E-40 requirement 5.5.3.2 clause c (currently reported in a note in section 7.1.6)

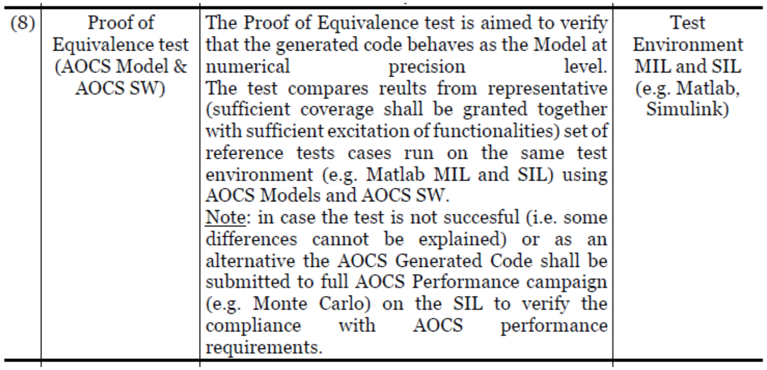
Extended Working Group – comments (3/5)

Autocoding Process: MC campaign on MIL & ‘Proof of Equivalence’ test SIL == MIL

What about running the MC campaign with the final SW (=SIL and not prototyped)?



Development Process with Autocoding
nt
AOC Code Generation and AOC Performance Validation
(8)
Comparison of results
(AOCS Model vs AOCS Code)
-- NUMERICAL EQUIVALENCE --
(6)
(8)
AOCS Code
Automated Unit
Testing (UT)
AOCS Performance
Verification
REPRESENTATIVE Tests
(AOCS Code)
(5)



(8)	Proof of Equivalence test (AOCS Model & AOCS SW)	The Proof of Equivalence test is aimed to verify that the generated code behaves as the Model at numerical precision level. The test compares reults from representative (sufficient coverage shall be granted together with sufficient excitation of functionalities) set of reference tests cases run on the same test environment (e.g. Matlab MIL and SIL) using AOCS Models and AOCS SW. Note: in case the test is not succesful (i.e. some differences cannot be explained) or as an alternative the AOCS Generated Code shall be submitted to full AOCS Performance campaign (e.g. Monte Carlo) on the SIL to verify the compliance with AOCS performance requirements.	Test Environment MIL and SIL (e.g. Matlab, Simulink)

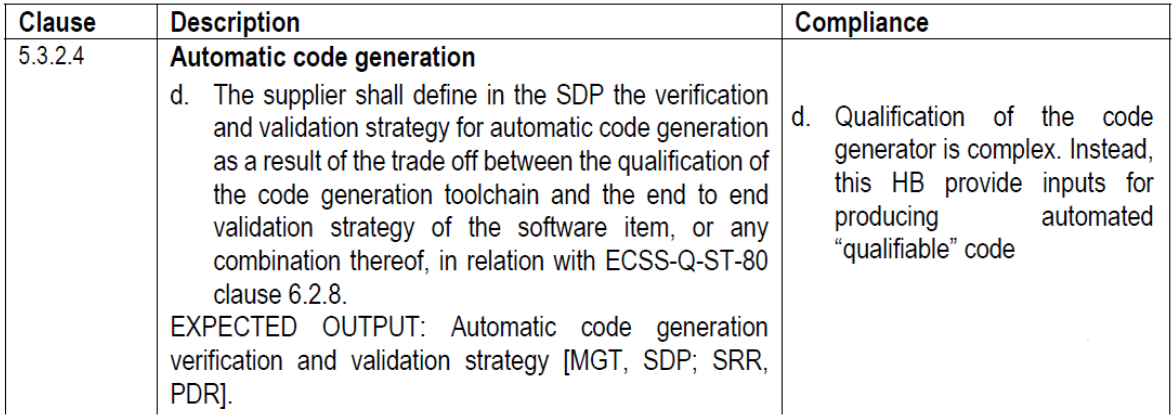
Despite it is considered against one of the advantages of the autocoding process, alternative verification approach where performance are verified in SIL will be included in the HB.

Even with such alternative approach any change needed from test results shall be implemented at model level.

Extended Working Group – comments (4/5)

V&V philosophy for Autocoding: Qualification of the tool-chain (ECSS E40 5.3.2.4.d)

Why the final product qualification cannot be achieved by qualification of the Autocoding generation tool-chain?



Clause	Description	Compliance
5.3.2.4	**Automatic code generation** d. The supplier shall define in the SDP the verification and validation strategy for automatic code generation as a result of the trade off between the qualification of the code generation toolchain and the end to end validation strategy of the software item, or any combination thereof, in relation with ECSS-Q-ST-80 clause 6.2.8. EXPECTED OUTPUT: Automatic code generation verification and validation strategy [MGT, SDP; SRR, PDR].	d. Qualification of the code generator is complex. Instead, this HB provide inputs for producing automated “qualifiable” code

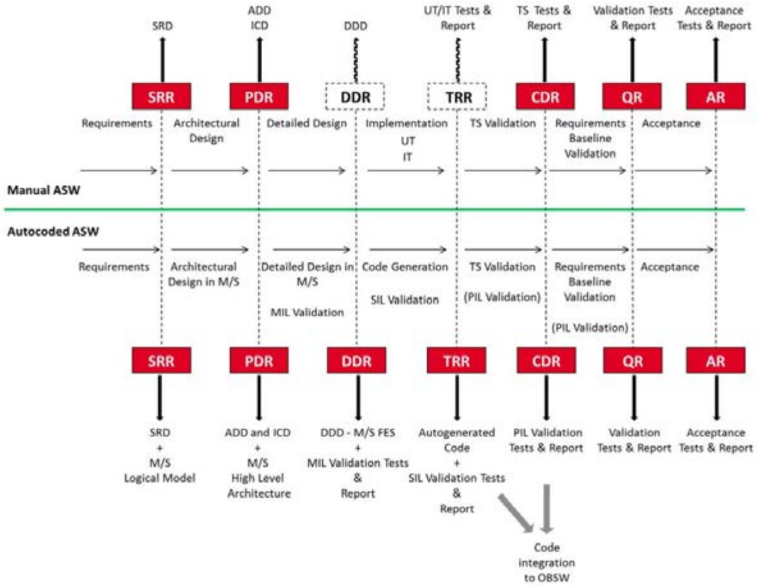
It is not intended to qualify the tool chain.

There are different reasons, the main one linked to the feasibility of this qualification considering the (quick) evolution of the tools, their complexity (all functions would need qualification) and the accessibility to the source code.

Anyway the compiler would not be qualified and the recommended approach has always been to qualify the final product.

Extended Working Group – comments (5/5)

V&V philosophy for Autocoding: deliverables, *"…implies that the GNC Model shall be delivered as part of the GNC code release note"*
Why the Model is a deliverable if it has been proved that the HB modelling guidelines are respected?



With the Autocoding the GNC Models became the TS for the SW generation and as such they have to be delivered to customer, to check verification, despite it is an automated process.

The maintainability of the code modifying the Simulink model (no manual change of flight code) is possible only if the model has been developed following the guidelines.

Extended Working Group – clarifications

Generic process defined by industry might differ on some specific points despite is generally closed to the HB process.

The HB will define the process (shared within the extended working group) that, if followed, will avoid many iterations within specific project reviews.
Any proposed deviation will be discussed.
The Extended Working Group iterations aim to clarify the needs and harmonize understandings, with the scope to minimize specific discussions

The Handbook today covers only AOCS Flight SW

When the working group has been established within SAVOIR cappella, the scope was to define the guidelines and the process for the AOCS Flight SW Autocoding, being recognized as the most suitable considering the use of specific environments and tools for development.
In a second phase it has been agreed to look forward to extend the concept to other OBSW elements

Extended Working Group – timeline

Planning

The following schedule is in place (target dates – number of meetings is TBC):

- 11/2019 **Replies to comments** distributed back to authors
- 12/2019 Invitation to meetings (Skype):
  - *01 2020 Meeting#1 – individual review meetings*
  - *02 2020 Meeting#2 – group review meeting(s)*
- 02/2020 **Wrap Up** finalization of the Handbook
- March 2020 **Final issue 1**

Q1 2020

Guidelines for the Automatic Code Generation for AOCS/GNC Flight SW Handbook

Conclusions