MathWorks®

Accelerating the pace of engineering and science

ECSS Workflow for Automatic Code Generation - ESA Software Product Assurance Workshop

REQUIREMENTS

Req. Baseline

ECSS-Q-ST-80C → 7.2 Product quality requirements → 7.2.1.3 (a)
a. For each requirement the method for verification and validation shall be specified.
EXPECTED OUTPUT: The following outputs are expected:
a. Requirement baseline [RB, SSS; SRR];
b. Technical specification [TS, SRS; PDR].

Design, Source Code and & Test Cases Traceability

ECSS-Q-ST-80C → 6.3.5 Testing and validation → 6.3.5.3 (a)
The supplier shall ensure through internal review that the test procedures and data are adequate, feasible and traceable and that they satisfy the requirements.
EXPECTED OUTPUT: Software product assurance reports [PAF, -; -]

Simulink Requirements

ECSS-Q-ST-80C – 6.2.6.5 (a)
Software containing deactivated code shall be verified specifically to ensure that the deactivated code cannot be activated or that its accidental activation cannot harm the operation of the system.
EXPECTED OUTPUT: Software verification report [DJF, SVR; CDR, QR, AR].

ECSS-Q-ST-80C - 7.2.1 Requirements baseline and technical specification

7.2.1.1 (a) The software quality requirements shall be documented in the requirements baseline and technical specification.
EXPECTED OUTPUT: The following outputs are expected:
a. Requirement baseline [RB, SSS; SRR];
b. Technical specification [TS, SRS; PDR].

7.2.1.2 (a) - The software requirements shall be:
1. correct;
2. unambiguous;
3. complete;
4. consistent;
5. verifiable;
6. traceable.

Design constraints (Equivalence classes, Boundary Values, Derived Requirements)

Simulation Cases Results

Test Cases Traceability

Coverage Metrics

Simulink Coverage

Simulink Design Verifier

Design Error Detection and Property Proving

Design Traceability

Automatic Test Case Generation

ECSS-E-ST-40C - 5.8.3.5 Verification of code
The supplier shall verify source code robustness (e.g. resource sharing, division by zero, pointers, run-time errors).
AIM: use static analysis for the errors that are difficult to detect at runtime.
EXPECTED OUTPUT: Robustness verification report [DJF, SVR; CDR]

ECSS-E-ST-40C - 5.8.3.13 Behaviour modelling verification

a. As support to the verification of the software requirements, the supplier shall verify the software behaviour using the behavioural view of the logical model produced in 5.4.2.3c.
EXPECTED OUTPUT: Software behaviour verification [DJF, SVR; PDR].

b. As support to the verification of the software architectural design, the supplier shall verify the software behaviour using the behavioural view of the architecture produced in clause 5.4.3.4
EXPECTED OUTPUT: Software behaviour verification [DJF, SVR; PDR].
c. As support to the verification of the software detailed design, the supplier shall verify the software behaviour using the software behavioural design model produced in 5.5.2.3a. eo c., by means of the techniques defined in 5.5.2.6.
EXPECTED OUTPUT: Software behaviour verification [DJF, SVR;CDR].

Requirement Based Functional Test Cases + Test Cases identified using Formal Methods

ECSS-E-ST-40C - 5.7.3.5 Evaluation of acceptance testing
a. The acceptance tests shall be traced to the requirements baseline.
EXPECTED OUTPUT: Traceability of acceptance tests to the requirements baseline [DJF, SVR; AR].

ECSS-Q-ST-80C – 6.2.6.1 (a) Activities for the verification of the quality requirements shall be specified in the definition of the verification plan.
NOTE: Verification includes various techniques such as review, inspection, testing, walk-through, cross-reading, desk-checking, model simulation, and many types of analysis such as traceability analysis, formal proof or fault tree analysis
EXPECTED OUTPUT: Software verification plan [DJF, SVerP; PDR].

Testing Environment Settings

Simulink Test

Simulation in the Loop Functional Testing

MODEL

SWRS + SADD

Model Coverage Analysis

Doc Templates Scripts

Simulink Report Generator

- Software Development Plan (SDP)
  - Modelling and Design Standards
- SW Requirements Specification (SRS)
  - Software Logical Model
  - Req. Traceability Matrix
- SW Design Document (SDD)
  - Software Architectural Design
  - Software Behaviour
  - Internal Interface Design
- Software Verification Report (SVR)
  - Autocode input model review
  - Test Traceability Matrix
  - Model Conformance Report
  - Model Coverage Report

...

ECSS-Q-ST-80C – 6.2.8.4 (a) Modelling standards for automatic code generation tools shall be defined and applied.
EXPECTED OUTPUT: Modelling standards [PAF, -; SRR, PDR].

Modelling Standards

Simulink Checks

Model Conformance Checks

ECSS-Q-ST-80C – 6.2.8.5 (a)
Adherence to modelling standards shall be verified.
EXPECTED OUTPUT: Software product assurance reports [PAF, -; -].

ECSS-E-ST-40C – 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].

Source Code Traceability

Autocoding Settings

ECSS-E-ST-40C – 5.8.3.5 Verification of code (a) The supplier shall verify the software code ensuring that:
1. the code is externally consistent with the requirements and design of the software item;
3. the code is traceable to design and requirements, testable, correct, and in conformity to software requirements and coding standards;
4. the code that is not traced to the units is justified;

EXPECTED OUTPUT: The following outputs are expected: (a) SW code traceability matrices [DJF, SVR;CDR]; (b) SW code verification report [DJF, SVR;CDR].

ECSS-Q-ST-80C - 6.2.8 Automatic Code Generation
6.2.8.1 (a) For the selection of tools for automatic code generation, the supplier shall evaluate the following aspects:
1. evolution of the tools in relation to the tools that use the generated code as an input;
2. customization of the tools to comply with project standards;
3. portability requirements for the generated code;
4. collection of the required design and code metrics;
5. verification of software components containing generated code;
6. configuration control of the tools including the parameters for customisation;
7. compliance with open standards.

6.2.8.3 (a) The required level of verification and validation of the automatic generation tool shall be at least the same as the one required for the generated code, if the tool is used to skip verification or testing activities on the target code.

Embedded Coder

ECSS-Q-ST-80C
6.2.3 Handling of critical software → 6.2.3.4 (a)
The supplier shall define, justify and apply measures to assure the dependability and safety of critical software. NOTE These measures can include:
- use of software design or methods that have performed successfully in a similar application;
- use of a "safe subset" of programming language;
- full inspection of source code;
- witnessed or independent testing;

6.2.3 Handling of critical software → 6.2.3.4 (a)
Reviews and inspections shall be carried out according to defined criteria, and according to the defined level of independence of the reviewer from the author of the reviewed item.

6.2.3 Handling of critical software → 6.2.3.4 (a)
Each review and inspection shall be based on a written plan or procedure.

Simulink Requirements

Source Code Traceability

Automatic Code Generation

Simulink Code Inspector

Automatic Code Inspection - structure and traceability

ECSS-Q-ST-80C – 6.3.4 Coding → 6.3.4.6
a. The supplier shall define measurements, criteria and tools to ensure that the software code meets the quality requirements.
EXPECTED OUTPUT: Software product assurance plan [PAF, SPAP; PDR].
b. The code evaluation shall be performed in parallel with the coding process, in order to provide feedback to the software programmers.

ECSS-Q-ST-80C - 6.3.4.1 (a) Coding standards (including consistent naming conventions and adequate commentary rules) shall be specified and observed.
EXPECTED OUTPUT: Coding standards [PAF, -; PDR].

Coding Standards → **Polyspace Bug Finder** → Code Conformance (MISRA,...)

ECSS-E-ST-40C - 5.8.3.5 Verification of Code
a. The supplier shall verify the software code ensuring that:
7. the effects of run–time errors are controlled;
8. there are no memory leaks;
9. numerical protection mechanisms are implemented

EXPECTED OUTPUT: The following outputs are expected:
(a) Software code traceability matrices [DJF, SVR; CDR]; (b) Software code verification report [DJF, SVR; CDR].
f. The supplier shall verify source code robustness (e.g. resource sharing, division by zero, pointers, run-time errors).
AIM: use static analysis for the errors that are difficult to detect at runtime.
EXPECTED OUTPUT: Robustness verification report [DJF, SVR; CDR]

Validation Objectives Settings → **Polyspace Code Prover** → Prove Absence of RT Errors

Doc Templates Scripts

Simulink Report Generator

- Software Development Plan (SDP)
  - Coding standards
  - Test Traceability Matrix
  - Tool Validation Documentation
- Software Unit Integration Test Plan (SUITP)
- Software Validation Report (SVR)
  - Code Generation Report
  - Software Unit Test Report
  - Software Code Traceability Matrix
  - Code Coverage Report
  - Robustness Report
  - Independent SW Validation Report
  - Review and Inspection Reports

...

Software Validation Test Cases Traceability

Software in the Loop (SIL) Unit Testing

SOURCE CODE

SIL Test Cases Results

Testing Environment Settings

Simulink Coverage

Code Coverage

Reuse of the Previous Test Cases Adding RT specific Test Cases

Simulink Test

Compiler

Code coverage versus criticality category	A	B	C	D
Source code statement coverage	100%	100%	AM	AM
Source code decision coverage	100%	100%	AM	AM
Source code modified condition and decision coverage	100%	AM	AM	AM
NOTE: "AM" means that the value is agreed with the customer and measured as per ECSS-Q-ST-80 clause 6.3.5.2.				

ECSS-Q-ST-80C - 6.2.8.2 (a) The requirements on testing applicable to the automatically generated code shall ensure the achievement of the same objectives as those for manually generated code.
EXPECTED OUTPUT: Validation and testing documentation [DJF, SValP; PDR], [DJF, SVS; CDR, QR, AR], [DJF, SUITP; PDR, CDR].

ECSS-E-ST-40C – 5.5.3.2 Software Unit Testing
a. The supplier shall develop and document the test procedures and data for testing each software unit.
EXPECTED OUTPUT: The following outputs are expected:
a. Software component design document and code (update) [DDF, SDD, source code; CDR];
b. Software unit test plan (update) [DJF, SUITP; CDR].

c. The unit test shall exercise:
1. code using boundaries at n-1, n, n+1 including looping instructions, while, for and tests that use comparisons;
2. all the messages and error cases defined in the design document;
3. the access of all global variables as specified in the design doc;
4. out of range values for input data, including values that can cause erroneous results in mathematical functions;
5. the software at the limits of its requirements (stress testing).

EXPECTED OUTPUT: Software unit test reports [DJF, - ; CDR].

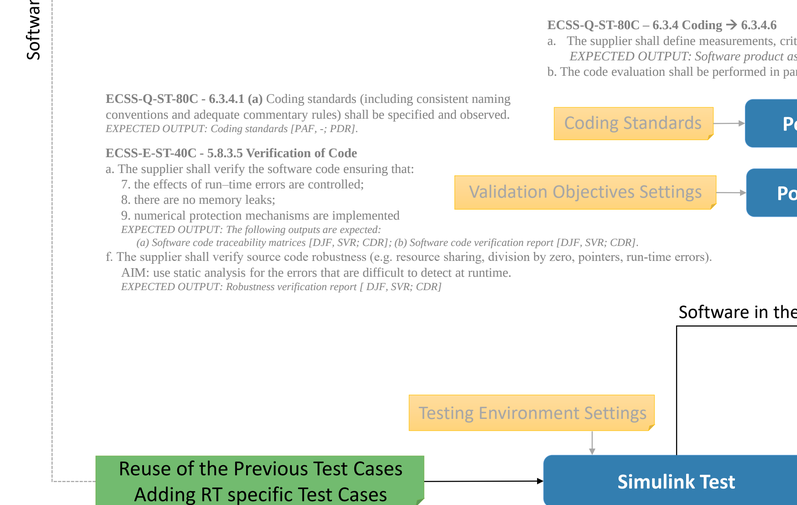
ECSS-E-ST-40C – 5.5.3.2 Software Unit Testing (b) The supplier shall test each software unit ensuring that it satisfies its requirements and document the test results.
EXPECTED OUTPUT: The following outputs are expected:
(a) Software component design document and code (update) [DDF, SDD, source code; CDR];
(b) Software unit test reports [DJF, - ; CDR].

EXECUTABLE OBJECT CODE

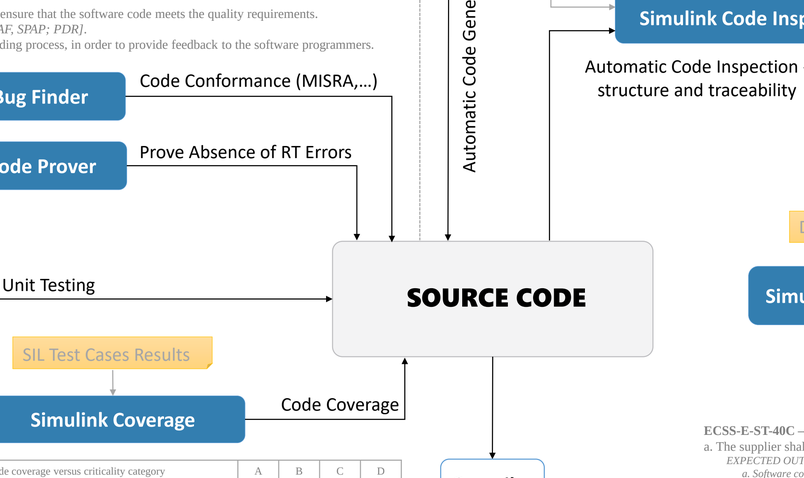
Processor and Hardware in the Loop (PIL and HIL) Unit Testing

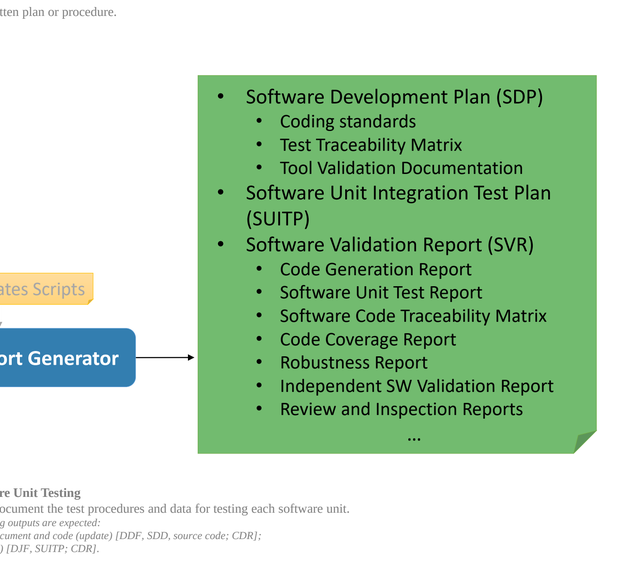
ECSS-Q-ST-80C – 6.3.5 Testing and validation → 6.3.5.29 (a) The validation shall include testing in the different configurations possible or in a representative set of them when it is evident that the number of possible configurations is too high to allow validation in all of them.
EXPECTED OUTPUT: Test and validation documentation [DJF, SValP; PDR], [DJF, SVS; CDR, QR, AR].

Effort Distribution in Traditional Development Workflows



Effort Distribution in Model-Based Design Workflows





MATLAB® & SIMULINK®

Prepared by: Albert Ramirez Perez, MathWorks
Albert.RamirezPerez@MathWorks.com
+49-89-45235-6772

September 2017