EUROPEAN COOPERATION

ECSS

FOR SPACE STANDARDIZATION

Space engineering

Software

ECSS Secretariat
ESA-ESTEC
Requirements & Standards Section
Noordwijk, The Netherlands

Foreword

ECSS is a cooperative effort of the European Space Agency, national space agencies and European industry associations for the purpose of developing and maintaining common standards. Requirements in this Standard are defined in terms of what shall be accomplished, rather than in terms of how to organize and perform the necessary work. This allows existing organizational structures and methods to be applied where they are effective, and for the structures and methods to evolve as necessary without rewriting the standards.

This Standard has been prepared by the ECSS-E-ST-40C Rev.1 Working Group, reviewed by the ECSS Executive Secretariat and approved by the ECSS Technical Authority.

Disclaimer

ECSS does not provide any warranty whatsoever, whether expressed, implied, or statutory, including, but not limited to, any warranty of merchantability or fitness for a particular purpose or any warranty that the contents of the item are error-free. In no respect shall ECSS incur any liability for any damages, including, but not limited to, direct, indirect, special, or consequential damages arising out of, resulting from, or in any way connected to the use of this Standard, whether or not based upon warranty, business agreement, tort, or otherwise; whether or not injury was sustained by persons or property or otherwise; and whether or not loss was sustained from, or arose out of, the results of, the item, or any services that may be provided by ECSS.

Published by: ESA Requirements and Standards Section
ESTEC, P.O. Box 299,
2200 AG Noordwijk
The Netherlands

Change log

ECSS-E-40A 19 April 1996	First issue
ECSS-E-40 Part1B 28 November 2003 ECSS-E-40 Part2B 31 March 2005	Second issue
ECSS-E-ST-40C 6 March 2009	Third issue
ECSS-E-ST-40C Rev.1 30 April 2025	Third issue Revision 1 Changes with respect to ECSS-E-ST-40C (6 March 2009) are identified with revision tracking. Main changes are: • Implementation of Change requests • Synchronization with other working groups: ECSS-E-ST-20-40C (ASIC, FPGA and IP Core engineering), ECSS-Q-ST-80C Rev2 (Software product assurance), ECSS-E-ST-80C (Security in space systems lifecycles) • Definitions updated • Annex A Table A-1: Security File added • Annex P.2.1<6>: Review objectives updated • Annex Q: Tables updated to take into account new requirements • Annex T SMP DRD added and SMP expected outputs identified in 5.2.10, • Annex U Software code verification added. Detailed changes Added requirements: 5.2.2.1b; 5.2.2.1c; 5.2.2.1d; 5.2.2.4a; 5.2.3.1b; 5.2.3.3b; 5.2.4.3b; 5.2.4.4b; 5.2.4.9b; 5.2.4.9c; 5.2.4.10a; 5.2.4.11a; 5.3.3.1b; 5.3.5.3a; 5.3.5.4a; 5.3.8.2b; 5.4.3.2b; 5.6.2.1d; 5.6.5a; 5.7.2.1b; 5.7.2.3b; 5.8.2.1e; 5.8.3.11d; 5.9.2.3b; 5.9.4.3a; 5.10.2.1f; 5.11.2a; 5.11.2b; 5.11.2c; 5.11.2d; 5.11.3a; 5.11.3b; 5.11.3c; 5.11.3d; 5.11.3e; 5.11.3f; 5.11.3g; 5.11.3h; 5.11.3i; 5.11.3j; 5.11.3k; 5.11.3l; 5.11.4a; 5.11.4b; 5.11.4c; 5.11.4d; 5.11.4e; 5.11.4f; 5.11.5.1a; 5.11.5.1b; 5.11.5.1c; 5.11.5.1d; 5.11.5.2a; 5.11.5.2b; 5.11.5.2c; 5.11.5.2d; 5.11.5.3a; 5.11.5.3b; 5.11.5.3c; 5.11.5.3d; 5.11.5.4a; 5.11.5.4b; 5.11.5.4c; 5.11.5.4d; 5.11.5.5a; 5.11.5.5b; 5.11.5.5c; 5.11.5.6a; 5.11.5.6b; 5.11.5.6c; 5.11.5.6d; 5.11.5.6e; 5.11.5.6f; 5.11.5.6g; 5.11.5.6h; 5.11.5.6i; B.2.1<5.14>a; B.2.1<5.15>a; B.2.1<6.1>b; H.2.1<9.9>a; M.2.1<7>a; O.2.1<5.5.3>c; T.2.1<1>a; T.2.1<2>a; T.2.1<3>a; T.2.1<4>a; T.2.1<4>b;

	T.2.1<4>c; T.2.1<5>a; T.2.1<5>b; T.2.1<6.1>a; T.2.1<6.1>b; T.2.1<6.2>a; T.2.1<6.3>a; T.2.1<6.4>a; T.2.1<6.5>a; T.2.1<7.1>a; T.2.1<7.2>a; T.2.1<7.3>a; T.2.1<7.4>a; T.2.1<8>a; T.2.1<9.1>a; T.2.1<9.2>a; T.2.1<10.1>a; T.2.1<10.1>b; T.2.1<10.1>c; T.2.1<10.2>a; T.2.1<10.3>a; T.2.1<10.4>a; T.2.1<10.5>a; T.2.1<10.6>a; T.2.1<10.6>b; T.2.1<11>a; T.2.1<12>a; T.2.1<12>b; T.2.1<13>a; T.2.1<14>a; T.2.1<14>b; T.2.1<14>c; T.2.1<15>a Modified requirements: 5.2.2.1a, 5.2.4.7a, 5.3.2.4b, 5.3.3.1a, 5.3.3.3c, 5.3.5.2a, 5.3.6.2a, 5.3.8.1a, 5.3.8.2a, 5.4.2.1a EO j, 5.6.2.2b, 5.6.3.1a, 5.6.3.2a, 5.8.2.2b, 5.8.3.1a, 5.8.3.5a, 5.8.3.5b, 5.8.3.5c, 5.8.3.6a, 5.8.3.8a, 5.8.3.12a, 5.8.3.12b, 5.8.3.12c, 5.9.2.1a, 5.9.3.3a, 5.9.4.1a, 5.10.4.3a, 5.10.6.1a, 5.10.7.1a, B.2.1<3>a, C.2.1<3>a, C.2.1<5.3>a, D.2.1<3>a, D.2.1<5.8>a, E.2.1<3>a, F.2.1<3>a, F.2.1<4.1>a, F.2.1<4.6>, F.2.1<4.7>a, G.2.1<3>a, H.2.1<3>a, H.2.1<7.2>a, H.2.1<9.5>a, H.2.1<10.2>a, I.2.1<3>a, J.2.1<3>a, K.2.1<3>a, L.2.1<3>a, M.2.1<3>a, M.2.1<5.2>c, M.2.1<5.2>d, N.2.1<3>a, N.2.1<4>b, N.2.1<5>b, O.2.1<3>a, O.2.1<4.3>a, O.2.1<5.3>e, P.2.1<3>a, O.2.1<5>b, Deleted requirements: None Editorial corrections: • Notes placed always after the text of the normative provision or Expected Output. • Examples (e.g.) moved from requirement text to a NOTE attached to the requirement. • Additional NOTE attached to some requirements.

Table of contents

Figures

Tables

Introduction

This Standard defines the principles and requirements applicable to space software engineering. ECSS-Q-ST-80 defines the principles and requirements applicable to space software product assurance.

The formulation of this Standard takes into account the existing ISO 9000 family of documents, and the ISO/IEC 12207 standard.

1 Scope

This software engineering Standard concerns the “product software”, i.e. software that is part of a space system product tree and developed as part of a space project.

This Standard is applicable, to the extent defined by the tailoring process, to all the elements of a space system, including the space segment, the launch service segment and the ground segment.

This Standard covers all aspects of space software engineering including requirements definition, design, production, verification and validation, transfer, operations and maintenance.

It defines the scope of the space software engineering processes and its interfaces with management and product assurance, which are addressed in the Management (–M) and Product assurance (–Q) branches of the ECSS System, and explains how they apply in the software engineering processes.

This Standard reflects the specific methods used in space system developments, and the requirements for the software engineering processes in this context. Together with the requirements found in the other branches of the ECSS Standards, this Standard provides a coherent and complete framework for software engineering in a space project.

This Standard is intended to help the customers to formulate their requirements and suppliers to prepare their responses and to implement the work.

This Standard is not intended to replace textbook material on computer science or technology, and such material is avoided in this Standard. The readers and users of this Standard are assumed to possess general knowledge of computer science.

The scope of this Standard is the software developed as part of a space project, i.e. “Space system product software”. This Standard also applies to the development of non–deliverable software that affects the quality of the deliverable product.

This Standard may be tailored for the specific characteristics and constraints of a space project in conformance with ECSS-S-ST-00.

2
Normative references

The following normative documents contain provisions, which through reference in this text, constitute provisions of this ECSS Standard. For dated references, subsequent amendments to, or revision of any of these publications do not apply. However, parties to agreements based on this ECSS Standard are encouraged to investigate the possibility of applying the more recent editions of the normative documents indicated below. For undated references, the latest edition of the publication referred to applies.

ECSS-S-ST-00-01	ECSS system – Glossary of terms
ECSS-E-ST-10-06	Space engineering - Technical requirements specification
ECSS-E-ST-10-11	Space product assurance – Human factors engineering
ECSS-M-ST-10	Space project management – Project planning and implementation
ECSS-M-ST-10-01	Space project management – Organization and conduct of reviews
ECSS-M-ST-40	Space project management – Configuration and information management
ECSS-Q-ST-80	Space product assurance – Software product assurance

3 Terms, definitions and abbreviated terms

3.1 Terms from other standards

a. For the purpose of this Standard, the terms and definitions from ECSS-ST-00-01 apply.

3.2 Terms specific to the present standard

3.2.1 acceptance test

test of a system or functional unit usually performed by the customer on his premises after installation, with the participation of the supplier to ensure that the contractual requirements are met

NOTE Adapted from ISO/IEC 2382-20:1990

3.2.2 automatic code generation

generation of source code with a tool from a model

NOTE Code issued from a generative artificial intelligence is considererd as manually generated code.

3.2.3 code coverage

percentage of the software that has been executed (covered) by the test suite

3.2.4 condition

boolean expression not containing boolean operators

3.2.5 configurable code

code that is only intended to be executed in certain specific configurations of the software product

NOTE This can be achieved either by use of compilation/link directives, parameter configuration (e.g. in a configuration file or database), or by target computer environment (e.g. hardware pin selection).

3.2.6 COTS, MOTS software

for the purpose of this Standard, commercial-off-the-shelf and modified-off-the-shelf software for which evidence of use is available

3.2.7 critical software

software of criticality category A, B or C

NOTE See ECSS-Q-ST-80 Table D-1 – Software criticality categories.

3.2.8 deactivated code

code that is not intended to be executed in the target software product, or in the operational configuration of the target software product

NOTE Code related to defensive programming is not considered as deactivated code.

3.2.9 decision

boolean expression composed of conditions and zero or more boolean operators that are used in a control construct.

NOTE 1 For example: "if.....then .....else" or the "case" statement are control construct.

NOTE 2 A decision without a boolean operator is a condition.

NOTE 3 If a condition appears more than once in a decision, each occurrence is a distinct condition.

3.2.10 decision coverage

measure of the part of the program within which every point of entry and exit is invoked at least once and every decision has taken "true" and "false" values at least once.

NOTE Decision coverage includes, by definition, statement coverage.

3.2.11 existing software

any software developed outside the business agreement to which this Standard is applicable, including software from previous developments provided by the supplier, software from previous developments provided by the customer, COTS and MOTS software, freeware and open source software

3.2.12 integration testing

testing in which software components, hardware components, or both are combined and tested to evaluate the interaction between them

[IEEE 610.12:1990]

3.2.13 logical model

implementation-independent model of software items used to analyse and document software requirements

NOTE The software logical model makes the software requirements understandable as a whole and not just individually. Depending on the notation, modelling language or method that is used to express it, it can also be used for:

- verification of completeness,
- verification of consistency,
- proof or formal verification of some high-level properties,
- generation of test scenarios.

The model shows the software functionalities and can include behavioural views.

3.2.14 margin philosophy

rationale for margins allocated to the performance parameters and computer resources of a development, and the way to manage these margins during the execution of the project

3.2.15 metric

defined measurement method and the measurement scale

NOTE 1 Metrics can be internal or external, and direct or indirect.

NOTE 2 Metrics include methods for categorising qualitative data.

[ISO/IEC 9126-1:2001]

3.2.16 migration

porting of a software product to a new environment

3.2.17 modified condition and decision coverage

measure of the part of the program within which every point of entry and exit has been invoked at least once, every decision in the program has taken "true" and "false" values at least once, and each condition in a decision has been shown to independently affect that decision's outcome

[adapted from DO-178C]

NOTE A condition is shown to independently affect a decision's outcome by varying that condition while holding fixed all other possible conditions.

3.2.18 operational

for the purpose of this Standard, related to the software operation

NOTE It is not related to the spacecraft operation.

3.2.19 portability (a quality characteristic)

capability of software to be transferred from one environment to another

3.2.20 processing unit

function which is defined to execute software

NOTE 1 The term covers the hardware functions such as processing core included in Central Processing Unit (CPU), Graphical Processing Unit (GPU), Vision Processing Unit (VPU), Tensor Processing Unit (TPU), Neural Processing Unit (NPU), Physics Processing Unit (PPU), Digital Signal Processor (DSP), Image Signal Processor (ISP).

NOTE 2 In the context of SW engineering, it also covers the software processing units such as interpreters, emulators and virtual machines.

3.2.21 quality characteristics (software)

set of attributes of a software product by which its quality is described and evaluated

NOTE A software quality characteristic can have multiple levels of sub-characteristics.

3.2.22 quality model (software)

set of characteristics and the relationships between them which provide the basis for specifying quality requirements and evaluating quality

[ISO/IEC 9126-1:2001]

3.2.23 real-time

pertaining to a system or mode of operation in which computation is performed during the actual time that an external process occurs, in order that the computation results can be used to control, monitor, or respond in a timely manner to the external process

[IEEE 610.12:1990]

3.2.24 regression testing (software)

selective retesting of a system or component to verify that modifications have not caused unintended effects and that the system or component still complies with its specified requirements

[IEEE 610.12:1990]

3.2.25 reusability

degree to which a software unit or other work product can be used in more than one computer program or software system

[IEEE 610.12:1990]

3.2.26 robustness

degree to which a system or component can function correctly in the presence of invalid inputs or stressful environmental conditions

[IEEE 610.12:1990]

3.2.27 runtime error

program error that occurs while the program is running

NOTE 1 The term is often used in contrast to other types of program errors, such as syntax errors and compile time errors.

NOTE 2 The runtime error may lead to a software or system failure, i.e. unexpected or undesired behaviour of the system.

NOTE 3 A human mistake made in requirements specification, design specification or coding can result in a fault to be present (latent) in a software item. This hidden defect, under particular circumstances, can manifest itself as an error (a discrepancy between an expected value or action and the actual one) which, in turn, can lead to a failure, i.e. as an unexpected/unintended behaviour of the system.

3.2.28 singular input

input corresponding to a singularity of the function

3.2.29 software

set of instructions and data executed on a processing unit

NOTE 1: See 3.2.20 for the definition of processing unit.

NOTE 2: Some processing units only require data, e.g. configuration of state machines or configuration data of a neural network.

NOTE 3 Files using Hardware Description Languages (e.g. VHDL, Verilog, System-C) used to model ASICs or bit stream files used to programme FPGAs are not software.

3.2.30 software component

part of a software system

NOTE 1 Software component is used as a general term.

NOTE 2 Components can be assembled and decomposed to form new components. In the production activities, components are implemented as units, tasks or programs, any of which can be configuration items. This usage of the term is more general than in ANSI/IEEE parlance, which defines a component as a “basic part of a system or program”; in this Standard, components are not always “basic” as they can be decomposed.

3.2.31 software intensive system

space system in which the dominant part of the constituents are software elements

NOTE In such systems, subsystems consist mainly of software. For this type of system, the majority of interfaces are software-software interfaces.

3.2.32 software item

see “software product”

3.2.33 software observability

property of a system for which the values of status variables can be determined throughout observations of the output variables

3.2.34 software product

set of software, procedures, scripts, documentation and their associated data

NOTE The term “software item” is synomymous

3.2.35 software problem

condition of a software product that causes difficulty or uncertainty in the use of the software

[CMU/SEI-92-TR-022]

3.2.36 software product assurance

totality of activities, standards, controls and procedures in the lifetime of a software product which establishes confidence that the delivered software product, or software affecting the quality of the delivered product, conforms to customer requirements

3.2.37 software unit

atomic level software component that can be subjected to stand-alone testing

3.2.38 statement coverage

measure of the part of the program within which every executable source code statement has been invoked at least once

3.2.39 stress test

test that evaluates a system or software component at or beyond its required capabilities

3.2.40 test case

set of test inputs, execution conditions and expected results developed for a particular objective such as to exercise a particular program path or to verify compliance with a specified requirement

3.2.41 test design

documentation specifying the details of the test approach for a software feature or combination of software features and identifying associated tests

3.2.42 test procedure

detailed instructions for the set up, operation and evaluation of the results for a given test

3.2.43 test script

file containing a set of commands or instructions written in native format (computer or tool processable) in order to automate the execution of one or a combination of test procedures (and the associated evaluation of the results)

3.2.44 threat

potential cause of an unwanted incident, which can result in harm to a system or organization

[adapted from ISO/IEC 27000:2018]

3.2.45 unit test

test of individual software unit

3.2.46 unreachable code

code that cannot be executed due to design or coding error

3.2.47 usability (a quality characteristic)

capability of the software to be understood, learned, used and liked by the user, when used under specified conditions

3.2.48 validation

<CONTEXT: software> process to confirm that the requirements are correctly and completely implemented in the final product

NOTE The definition of validation at software level differs from the definition of validation at system level.

3.2.49 verification

<CONTEXT: software> process to confirm that adequate specifications and inputs exist for any activity, and that the outputs of the activities are correct and consistent with the specifications and input

NOTE The definition of verification at software level differs from the definition of verification at system level.

3.2.50 vulnerability

weakness which can be exploited by a threat source

3.2.51 walk-through

static analysis technique in which a designer or programmer leads members of the development team and other interested parties through a software product, and the participants ask questions and make comments about possible errors, violation of development standards, and other problems

[IEEE Std 1028-1997]

3.3 Abbreviated terms

For the purpose of this Standard and of ECSS-Q-ST-80, the abbreviated terms from ECSS-S-ST-00-01 and the following apply:

For the definition of DRD acronyms see Annex A.

NOTE The abbreviated terms are common for the ECSS-E-ST-40 and ECSS-Q-ST-80 Standards.

Abbreviation	Meaning
AR	acceptance review NOTE The term SW-AR can be used for clarity to denote ARs that solely involve software products.
ASIC	application-specific integrated circuit
CDR	critical design review NOTE The term SW-CDR can be used for clarity to denote CDRs that solely involve software products.
COTS	commercial-off-the-shelf
CPU	central processing unit NOTE The term CPU is commonly used to identify one or a group of processing units (PU).
DDF	design definition file
DDR	detailed design review
DJF	design justification file
DRD	document requirements definition
ECSS	European Cooperation for Space Standardization
eo	expected output
FPGA	field-programmable gate array
GS	ground segment
HMI	human machine interface
HSIA	hardware-software interaction analysis
HW	hardware
ICD	interface control document
IRD	interface requirements document

Abbreviation	Meaning
ISO	International Organization for Standardization
ISV	independent software validation
ISVV	independent software verification and validation
MGT	management file
MF	maintenance file
MOTS	modified off-the-shelf
OBCP	on-board control procedure
OP	operational plan
ORR	operational readiness review
OTS	off-the-shelf
PAF	product assurance file
PDR	preliminary design review NOTE The term SW-PDR can be used for clarity to denote PDRs that solely involve software products.
PRR	preliminary requirement review
PU	processing unit
QR	qualification review NOTE The term SW-QR can be used for clarity to denote QRs that solely involve software products.
RB	requirements baseline
SCMP	software configuration management plan
SDD	software design document
SDE	software development environment
SF	security file
SOS	software operation support
SPA	software product assurance
SPAMR	software product assurance milestone report
SPAP	software product assurance plan
SPR	software problem report
SRB	software review board
SRR	system requirements review NOTE The term SW-SRR can be used for clarity to denote SRRs that solely involve software products.
SSMP	software security management plan
SVSR	software validation specification review
SW	software
SWE	software engineering

Abbreviation	Meaning
SWRR	software requirements review
TRR	test readiness review
TS	technical specification

3.4 Nomenclature

The following nomenclature applies throughout this document:

a. The word "shall" is used in this Standard to express requirements. All the requirements are expressed with the word "shall".

b. The word "should" is used in this Standard to express recommendations. All the recommendations are expressed with the word "should".

NOTE It is expected that, during tailoring, recommendations in this document are either converted into requirements or tailored out.

c. The words "may" and "need not" are used in this Standard to express positive and negative permissions, respectively. All the positive permissions are expressed with the word "may". All the negative permissions are expressed with the words "need not".

d. The word "can" is used in this Standard to express capabilities or possibilities, and therefore, if not accompanied by one of the previous words, it implies descriptive text.

NOTE In ECSS "may" and "can" have completely different meanings: "may" is normative (permission), and "can" is descriptive.

e. The present and past tenses are used in this Standard to express statements of fact, and therefore they imply descriptive text.

4 Space system software engineering principles

4.1 Introduction

This clause 4 introduces the structure of this Standard and the framework of the space software engineering processes that form its basis.

The context of space software engineering is the overall space system engineering process. This Standard focuses on space software engineering processes requirements and their expected outputs. This clause 4 introduces the software engineering processes in their context, and the structure of the Standard.

Software is found at all levels, ranging from system functions down to low-level components interacting with hardware, including safety and mission critical functions. Therefore, a special emphasis is put in this Standard on the system-software relationship and on the verification and validation of software items.

This Standard is complemented by ECSS-Q-ST-80 Space product assurance — Software product assurance, which specifies the product assurance aspects and is the entry point for ECSS-E-ST-40 into the Q-series of standards. Requirements for space configuration and information management are in ECSS-M-ST-40, which includes the software DRD for software configuration file. Together, these standards either define or refer to the definition of all software relevant processes for space projects. ECSS-Q-ST-20 is the reference, through ECSS-Q-ST-80, for the software acquisition process, and the software management process tailors ECSS-M-ST-10 for software.

Figure 4-1 presents the different software related processes implemented by ECSS-E-ST-40, ECSS-Q-ST-80 and other ECSS Standards.

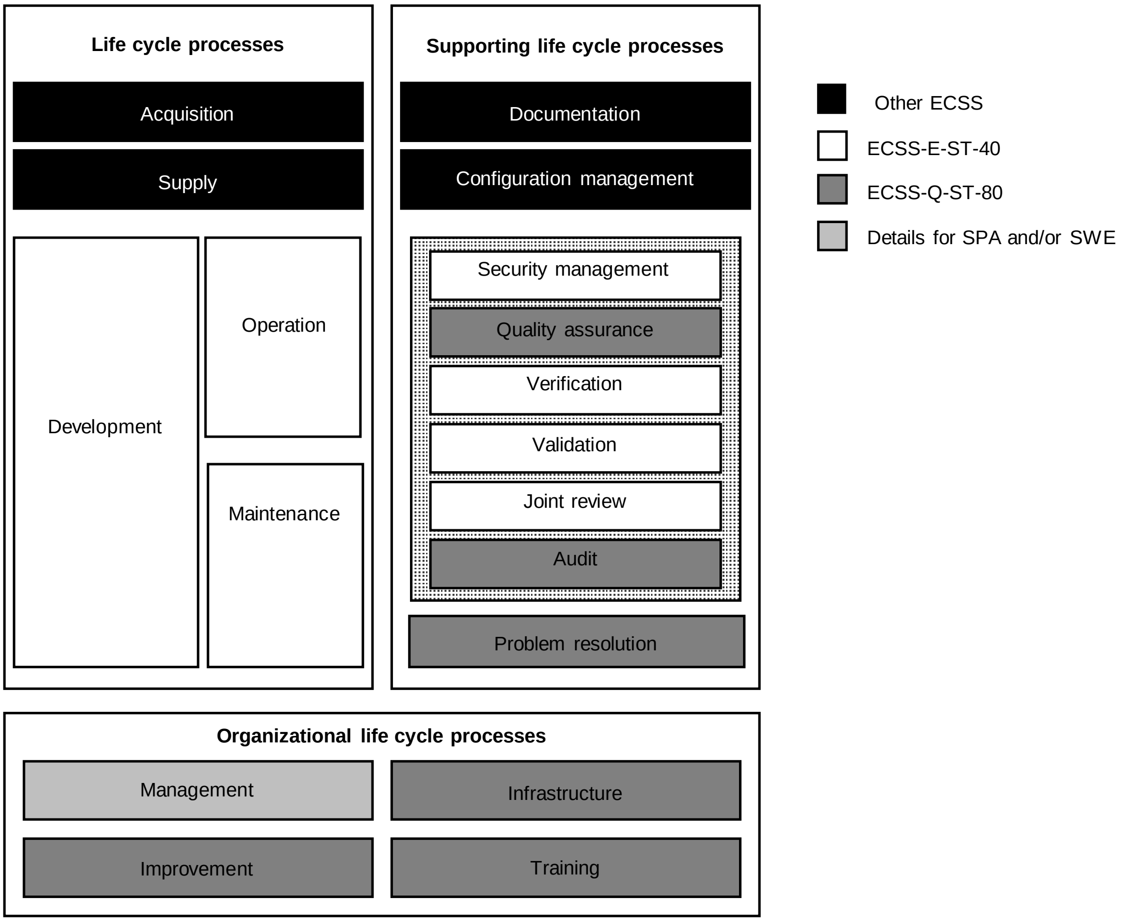

Figure 4-1: Software related processes in ECSS Standards

4.2 Overview of space system software engineering processes

4.2.1 General

In accordance with the ECSS theoretical concept, a fundamental principle of this Standard is the 'customer–supplier' relationship, assumed for all software developments. The project organization is defined in ECSS-M-ST-10. The customer is, in the general case, the procurer of two associated products: the hardware and the software for a system, subsystem, set, equipment or assembly. The concept of the "customer–supplier" relationship is applied recursively, i.e. the customer can be a supplier to a higher level in the space system. The software customer therefore has two important interfaces:

- the software customer interfaces with his software and hardware suppliers in order to adequately allocate to them functional and performance requirements, through the functional analysis.

- the software customer assumes a supplier role to interface in turn with his customer at the next higher level, ensuring that higher level system requirements are adequately taken into account.

The customer derives the functional and performance requirements for the hardware and software, based on system engineering principles and methods. The customer can also control the interface between the software and hardware. Software items are defined in the system breakdown at different levels. Nevertheless, it is important to manage the software–software interfaces irrespective of the level at which they occur. The customer's requirements are specified by this process, and they provide the starting point for the software engineering.

Reviews are the main interaction points between the customer and the supplier. They also synchronize software engineering processes. The reviews relevant to the software engineering processes are the SRR, PDR, CDR, QR, AR, and ORR as defined by ECSS-M-ST-10. This Standard offers in addition the possibility to anticipate the PDR in a SWRR, and the CDR into a DDR. The SWRR and DDR are executed as a first part of the ECSS-M-ST-10 reviews, but precede them.

All reviews are applicable to software. The reviews occur at different levels in the customer–supplier hierarchy and are sequenced according to the overall system level planning and selected life-cycle.

The notion of engineering processes is fundamental to this Standard. The processes provide the means to describe the overall constraints and interfaces to the engineering processes at system level. At the same time, they provide the possibility to the supplier to implement the individual activities and tasks implied by the processes in accordance with a selected software life cycle. This Standard is a process model, and does not prescribe a particular software life cycle. Although the spacecraft reviews suggest a waterfall model, the software development plan can implement any life cycle, as iterative model, spiral model, and Agile model or by the use of the technical reviews, less formal than the project reviews.

When the software development is included in a complete space project, the software engineering processes have also a relationship with the project phases (0, A, B, C, D, E, F) and in particular with the system level reviews (e.g. the system-software activities are executed in phase B). For other software developments which do not rely on a particular system, or which are developed for reuse, not applicable requirements can be tailored.

A software development plan defines and instantiates in detail the particular implementation of this standard in a project.

Figure 4-2 identifies the main concurrent software engineering processes and shows how and when they are synchronized by the customer/supplier reviews. Several iterations of software engineering processes can occur depending on the selected software life cycle.

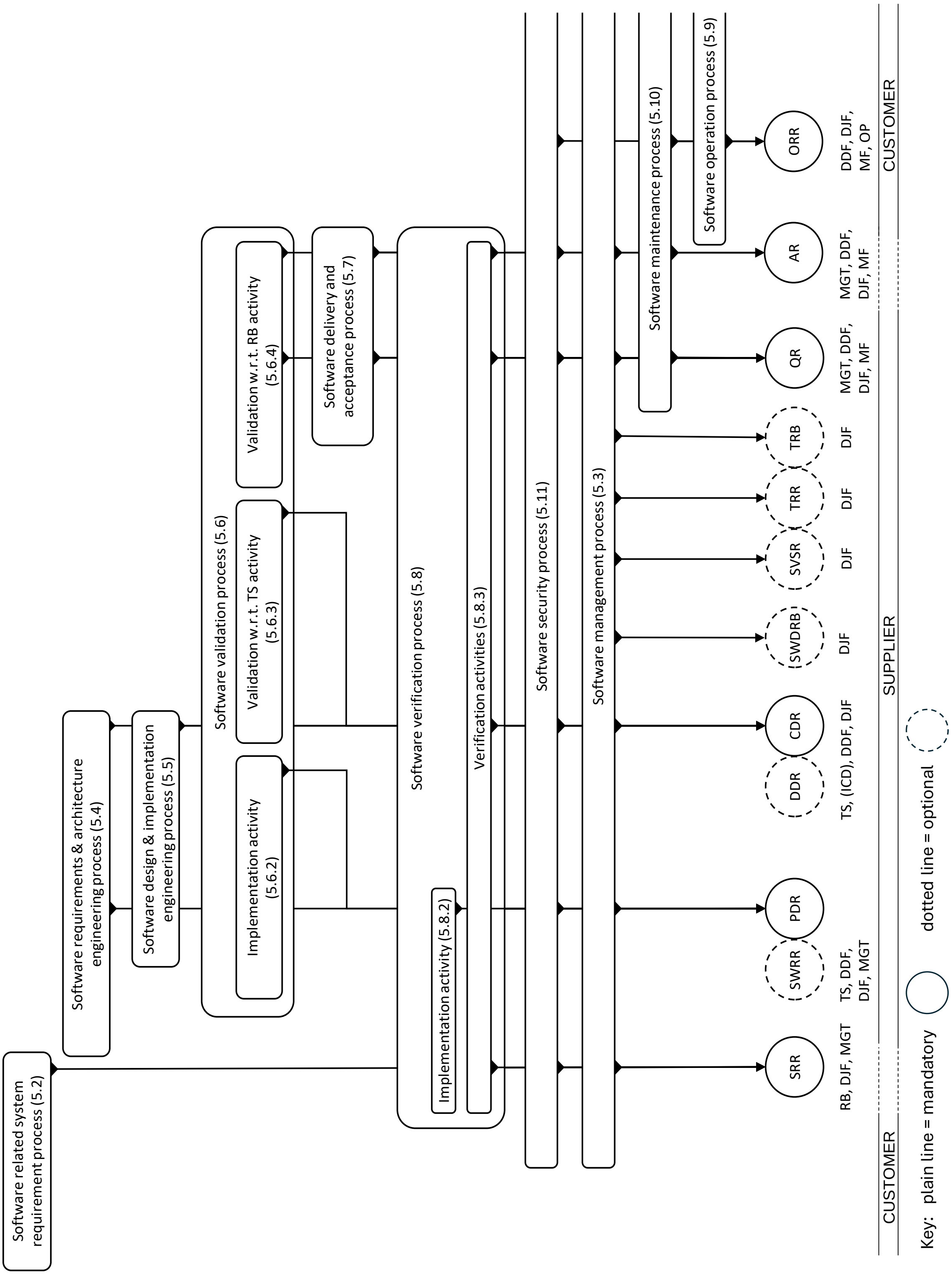
Software related system requirement process (5.2)
Software requirements & architecture engineering process (5.4)
Software design & implementation engineering process (5.5)
Software validation process (5.6)
Implementation activity (5.6.2)
Validation w.r.t. TS activity (5.6.3)
Validation w.r.t. RB activity (5.6.4)
Software delivery and acceptance process (5.7)
Software verification process (5.8)
Implementation activity (5.8.2)
Verification activities (5.8.3)
Software security process (5.11)
Software management process (5.3)
Software maintenance process (5.10)
Software operation process (5.9)
SRR
SWRR
PDR
DDR
CDR
SWDRB
SVSR
TRR
TRB
QR
AR
ORR
RB, DJF, MGT
TS, DDF, DJF, MGT
TS, (ICD), DDF, DJF
DJF
DJF
DJF
DJF
MGT, DDF, DJF, MF
MGT, DDF, DJF, MF
DDF, DJF, MF, OP
CUSTOMER
SUPPLIER
CUSTOMER
Key: plain line = mandatory
dotted line = optional

Figure 4-2: Overview of the software life cycle processes

4.2.2 Software related system requirements process

The software related system requirement process produces the information for input to the system requirements review (SRR). This establishes the functional and the performance requirements baseline (including the interface requirement specification) (RB) of the software development. It constitutes the link between the space system processes, as defined in ECSS-E-ST-10 and ECSS-E-ST-70, and the software processes.

According to the recursive customer-supplier model of ECSS, at each level, the customer is responsible for the delivery of a system in which the developed software is integrated. According to the recursive system-subsystem model of ECSS-E-ST-10, he is responsible for the specification of the system requirements at lower level and, in particular, for any software comprised in the system.

ECSS-E-ST-10 describes the following system engineering functions:

- requirement engineering, producing the system and lower level functional and technical specifications,
- system analysis, consolidating requirements through trade-off and various analysis (mission, requirement, functional, physical, performance), producing in particular the functional and physical architecture,
- system design (producing the physical architecture) and system configuration (producing the physical system with the software),
- system verification as being conformant to requirements, and
- system engineering integration and control, including in particular system integration, its relation with production, operation, product assurance and management, and the engineering database, the interface management and the technical budget management.

The system engineering functions giving input to the software are the requirement engineering, the system verification and the system engineering integration and control, and clause 5.2 is organised along this line. For software, the system analysis and design only support the system requirement engineering, the system configuration includes the software development.

According to ECSS-E-ST-10, the system engineering produces "lower level element functional specification" in phase A before the Preliminary Requirement Review (PRR), and "technical specification" of lower level elements in the preliminary definition for the system solution selected at end of Phase A and established in phase B (ECSS-E-ST-10C Rev.1 4.3.5a). For software, this document is the requirements baseline addressed in this clause.

4.2.3 Software management process

This process covers the complete life cycle of the software project.

The M-branch of the ECSS standards defines how to manage the software project. This process tailors the M standards for software-specific issues, in particular ECSS-M-ST-10. In addition, the software product assurance requirements specified in ECSS-Q-ST-80 are used for the control of space systems software projects. These requirements are not repeated here.

The main activity is the production of a software development plan including the life cycle description, activities description, milestones and outputs, the techniques to be used, and the risks identification.

The management process also includes the organization and handling of all the joint reviews, the definition of procedures for the management of the system interface, and the technical budget and margin management.

The reviews are defined and positioned in the life cycle in clause 5.3. They are called in their relevant process in clause 5.2, 5.4, 5.5, 5.6, and 5.7. The data package for each review is specified in Annex A.

Software requirements or detailed design dedicated reviews (i.e. SWRR and DDR) are defined as anticipation of the PDR and CDR respectively.

4.2.4 Software requirements and architecture engineering process

The software requirements and architecture engineering process consists of

- the elaboration of the technical specification, including the preliminary definition of the ICD (TS), which is the supplier's response to the requirements baseline including the interface requirement specification,
- the architectural design documented in a preliminary SDD.

It is the duty of the supplier to involve all stakeholders in the requirement elicitation.

As part of this process, the result of all significant trade-offs, feasibility analyses, make-or-buy decisions and supporting technical assessments are documented in a design justification file (DJF).

Preliminary design reviews (PDRs) are held to review the progress and outputs of the software requirements and architecture engineering process.

4.2.5 Software design and implementation engineering process

One of the outputs of this process is the detailed design of the software items identified in the software product tree (see ECSS-M-ST-10). It is provided in response to the technical specification, the ICD and the preliminary DDF. All elements of the software design are documented in the design definition file (DDF). The DDF contains all the levels of design engineering results, including software code listings.

The rationale for important design choices, and analysis and test data that show that the design meets all requirements, is added to the DJF by this process. Critical design reviews (CDRs) are held to review the progress and outputs of the software design and implementation engineering process.

As part of this process, the code, unit testing and integration testing of the software product are also produced. All elements of the testing activities are documented in the design justification file (DJF).

4.2.6 Software validation process

In this standard, the word software validation refers to software product testing against both the technical specification and the requirements baseline.

This process is intended to confirm that the technical specification and the requirements baseline functions and performances are correctly and completely implemented in the final product.

The result of this process is included in the DJF.

This process is established to produce a software validation plan (”process implementation”) that is reviewed before any validation activity starts. The plan includes the validation activity with respect to the technical specification (which is held before the CDR) and the validation activity with respect to the requirements baseline (which is held before the QR and possibly repeated at AR).

This process can include a software validation specification review (SVSR) before or combined with the TRR to ensure that the software validation specification is complete and gives confidence that the requirements are properly validated. The SVSR is particularly important for large projects to avoid any delay in the start of the software validation campaign.

This process can include a test readiness review (TRR) before the validation against the TS and the validation against the RB to verify that all test facilities and test cases and procedures are available and under configuration control before the software validation process starts (ECSS-Q-ST-80 requirement 6.1.5a) and possibly before each significant test campaign as defined in the software development plan.

The results relevant to the validation against the TS are checked at the CDR, and those against the RB are verified at the QR, using the DJF as input.

This process can be executed with varying degrees of independence. The degree of independence can range from the same person, or a different person in the same organization, to a person in a different organization, with varying degrees of separation. Where the software validation process and the software verification process are executed by an organization independent of the supplier, it is called Independent Software Verification and Validation (ISVV), or Independent Software Validation (ISV) if only the validation process is independent.

4.2.7 Software delivery and acceptance process

This process prepares the software product for delivery and testing in its operational environment. It is completed with an acceptance review (AR), with the DJF as input. The acceptance review is a formal event in which the software product is evaluated in its operational environment. It is carried out after the software product is transferred to the customer and installed on an operational basis.

4.2.8 Software verification process

The software verification process is intended to confirm that adequate specifications and inputs exist for every activity and that the outputs of the activities are correct and consistent with the specifications and inputs.

This process is concurrent with all the previous processes.

The customer verifies the requirements baseline for the SRR.

For the supplier, this process is established before the PDR to produce a software verification plan (”process implementation”). This process includes all the verification activities of all the expected outputs of the development activities

The result of this process is included in the DJF and reported at each review.

This process can be executed with varying degrees of independence. The degree of independence can range from the same person, or a different person in the same organization, to a person in a different organization, with varying degrees of separation. Where the software verification process and the software validation process are executed by an organization independent of the supplier, it is called Independent Software Verification and Validation (ISVV).

4.2.9 Software operation process

The operation process can start after completion of the acceptance review of the software. Since software products form an integrated part of a space system, the phasing and management of operations are determined by the overall system requirements and applied to the software products. The operation process is not directly connected to the overall mission phase E, but is instead determined by the requirement at system level to operate the software product at a given time.

Indeed, software operation in this Standard is totally different from spacecraft or payload operations performed by ground stations on space systems. Software operation in this Standard includes mainly the helpdesk and the link between the users, the developers or maintainers, and the customer.

The organisational entity responsible for the software operation support (SOS entity) ensures that the software remains operational for the users. Examples of SOS entities and users for different types of software are typically the following:

- for a space platform flight software:
  - the user is the spacecraft operation team when he uses the on-board software for commanding the spacecraft and receiving telemetry;
  - the SOS entity is the spacecraft operation team when he patches, dump or reboot the flight software.
- for manned mission flight software such as microgravity experiment software:
  - the user is the astronaut who manipulates the experiment through the laptop in space;
  - the SOS entity is either the astronaut or the ground centre, depending on who acts on the software for installations, reboot, patching, etc.

- for ground segment software:
  - the user are the spacecraft and ground segment operations team who uses the ground segment software to operate the spacecraft;
  - the SOS entity is the ground segment maintenance team who performs the installation and the deployment of the ground software.
- for payload data ground segment software:
  - the user is the operation team who uses the ground segment software to produce, quality control and disseminate data products;
  - the SOS entity is the ground segment maintenance team who performs the installation and the deployment of the ground software.
- for a software tool e.g. a distributed database with a human machine interface:
  - the user uses the software through the human machine interface;
  - the SOS entity administrates the database by supporting user's requests, rebooting or backing up the system.

4.2.10 Software maintenance process

The software maintenance process covers software product modification to code or associated documentation for correcting an error, a problem or implementing an improvement or adaptation. This process is activated before the software product is used in operations and usually after its qualification.

The maintenance process contains the activities and tasks of the maintainer. The objective is to modify an existing software product while preserving its functional integrity. This process includes the migration and retirement of the software product. The process ends with the retirement of the software product.

The activities provided in clause 5.10 are specific to the maintenance process; however, the process can utilize other processes in this Standard. If the software engineering processes are utilized, the term supplier there is interpreted as maintainer.

The maintainer manages the maintenance process at the project level following the management process, which is instantiated for software in this process.

Both the documents and the reviews identified by the clauses in 5.10 are part of the general maintenance activities for the space systems, and the requirements for these reviews and documentation are part of the space system maintenance engineering requirements, covered in other ECSS Standards. The provisions of clause 5.10 produce the required software engineering inputs for this system level activities.

4.2.11 Software security process

This process is performed throughout the full lifecycle of the software.

It is supported by a software security analysis that is systematically maintained at different points in the lifecycle of the software. The security analysis starts at system level (top level), i.e. where the software is part of a system. The software security analysis is based on the system level analysis. The security analysis is performed and maintained at each product architectural level of the system in a hierarchical manner. This implies that changes to the analysis at a given level have to be propagated to higher and lower levels to ensure continued consistency and validity. Along the lifecycle, the security analysis can use different methods and be extended in order to support the identification and treatment of risks but also to support other engineering processes, for example in the selection of design options or evaluation of validation results. The software security analysis is maintained along the lifecycle. It is reviewed periodically, at specific project milestones and on the basis of relevant events, e.g. change requests.

The software security analysis is used to ensure that security risks are properly addressed over the lifecycle of the software. It is also used to assess and drive the design, implementation and operation of secure software.

Although security risks can be associated in most cases to vulnerabilities there are also exceptions, e.g. risks introduced accidentally or unknowingly by inadequately trained users.

The software security process is concurrent and integrated with all the other processes described in this document and with risk management.

4.3 Organization of this Standard

This Standard contains one normative clause (clause 5) which is organized with respect to the software processes and activities breakdown described in Figure 4-3. Each process includes activities, which are themselves decomposed into a list of one single or several tasks in the shape of process requirements (clauses), producing expected outputs.

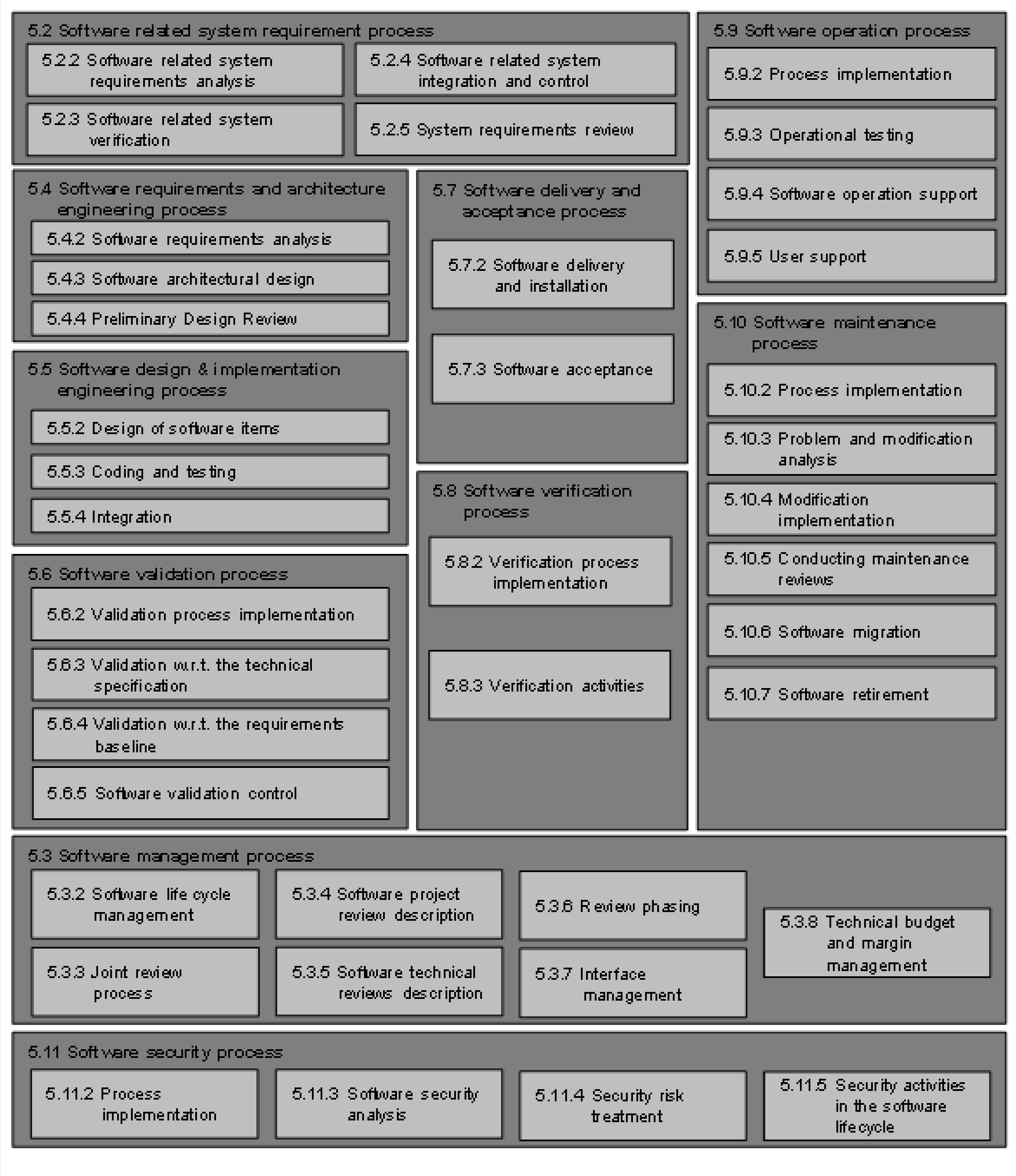
5.2 Software related system requirement process
5.2.2 Software related system requirements analysis
5.2.4 Software related system integration and control
5.2.3 Software related system verification
5.2.5 System requirements review
5.9 Software operation process
5.9.2 Process implementation
5.9.3 Operational testing
5.9.4 Software operation support
5.9.5 User support
5.4 Software requirements and architecture engineering process
5.4.2 Software requirements analysis
5.4.3 Software architectural design
5.4.4 Preliminary Design Review
5.7 Software delivery and acceptance process
5.7.2 Software delivery and installation
5.7.3 Software acceptance
5.5 Software design & implementation engineering process
5.5.2 Design of software items
5.5.3 Coding and testing
5.5.4 Integration
5.10 Software maintenance process
5.10.2 Process implementation
5.10.3 Problem and modification analysis
5.10.4 Modification implementation
5.10.5 Conducting maintenance reviews
5.10.6 Software migration
5.10.7 Software retirement
5.8 Software verification process
5.8.2 Verification process implementation
5.8.3 Verification activities
5.6 Software validation process
5.6.2 Validation process implementation
5.6.3 Validation w.r.t. the technical specification
5.6.4 Validation w.r.t. the requirements baseline
5.6.5 Software validation control
5.3 Software management process
5.3.2 Software life cycle management
5.3.4 Software project review description
5.3.6 Review phasing
5.3.8 Technical budget and margin management
5.3.3 Joint review process
5.3.5 Software technical reviews description
5.3.7 Interface management
5.11 Software security process
5.11.2 Process implementation
5.11.3 Software security analysis
5.11.4 Security risk treatment
5.11.5 Security activities in the software lifecycle

Figure 4-3: Structure of this Standard

The software documentation produced by all the expected output of this Standards is summarized in Annex A.

Annex B to Annex P and Annex T specify the DRD (the document requirements) allowing to organize most of the expected output into documents.

Annex Q shows, for each reviews, the documents (and the expected outputs without DRDs) that are delivered at the review, as well as the trace to the process requirements.

Annex R is a pre-tailoring of this Standard according to the criticality categories as defined in the ECSS-Q-ST-80.

Annex S gives other tailoring guidelines.

Annex T identifies recommended software code checks.

The Bibliography gives reference to other standards and documents.

4.4 Tailoring of this Standard

The general requirements for selection and tailoring of applicable standards are defined in ECSS-S-ST-00.

Several drivers for tailoring, include security, dependability and safety aspects, software development constraints, product quality objectives and business objectives.

Tailoring for dependability and safety aspects is based on the selection of requirements related to the verification, validation and levels of proof demanded by the criticality of the software. Annex R contains a tailoring of this Standard based on software criticality.

Tailoring for security is based on the selection of requirements related to the verification, validation and levels of proofs demanded by the sensitivity of the software.

Tailoring for software development constraints considers the special characteristics of the software being developed and of the development environment. The type of software development (e.g. database or real-time) and the target system (e.g. embedded processor, host system, programmable device, or application specific integrated circuits) is also taken into account (see Annex S of this Standard). Specific requirements for verification, review and inspection are imposed, for example, when full validation on the target computer is not feasible or where performance goals are difficult to achieve.

Tailoring for product quality and business objectives is done by selecting requirements on quality of the products as explained in clause 7 of ECSS-Q-ST-80. In this process, the customer specifies the quality objectives for the product.

4.5 Security aspects of this Standard

Given the particularities related to security requirements, the following consideration is made:

Security assurance requirements significantly influence the security requirements for software product development. Higher levels of security assurance increase confidence in the security features but demand increased security controls on the development and evaluation of the software, to limit the potential for weaknesses and vulnerabilities present in the software.

5 Requirements

5.1 Introduction

This clause 5 defines the requirements for engineering software for space systems, applicable to any space projects producing computer software.

Each requirement can be identified by a hierarchical number, made of the four digits of the clause (e.g. 1.2.3.4), followed by a letter (e.g. “a” or “b”) if the clause needs several requirements.

The text of the requirement is followed, where necessary, by further explanation of the aim (normative) or of a note (informative).

For each requirement, the associated output is given in the “expected output” section. When several outputs are expected, they are identified by a letter in italic (e.g. “*a*” or “*b*”). With each output, the destination file of the output is indicated in brackets, together with the corresponding document DRD (after a comma) and review (after a semicolon).

Example 1:

> **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].*

denote the output *a* of the requirement 5.5.3.2a. contained in the SDD and the source code documents, part of the design definition file, requested for the CDR review, and the output b of the requirement 5.5.3.2a. contained in the SUITP document, part of the design justification file, requested for the CDR review.

Example 2:

> **5.10.5.2 Baseline for change**
>
> a. Upon successful completion of the reviews, a baseline for the change shall be established.
>
> *EXPECTED OUTPUT: Baseline for changes [MF, - ; -]*

denote the output of requirement 5.10.5.2a. contained in the maintenance file, while no DRD nor reviews are specified.

The list of DRDs is given in Annex A.

Some requirements depend on the nature of the software. This is explicitly mentioned in the requirements. They include for example flight or ground software, presence or not of a HMI, independence or not of the V&V, real-time or not.

5.2 Software related system requirement process

5.2.1 Overview

The software related system requirement process consists of the following activities:

- software related system requirements analysis;
- software related system verification;
- software related system integration and control.

5.2.2 Software related system requirements analysis

5.2.2.1 Specification of system requirements allocated to software

ECSS-E-ST-40_0860001

a. The customer shall derive system requirements allocated to software from an analysis of the specific intended use of the system, from the results of the security analysis, from the results of the hardware/software co-engineering analysis, and from the results of the safety and dependability analysis.

EXPECTED OUTPUT: The following outputs are expected:

a. Functions and performance system requirements allocated to software [RB, SSS; SRR];

b. Verification and validation product requirements [RB, SSS; SRR];

c. *Software operations requirements [RB, SSS; SRR];*

d. *Software maintenance requirements [RB, SSS; SRR];*

e. *Requirements for in flight modification capabilities [RB, SSS; SRR];*

f. *Requirements for real-time [RB, SSS; SRR];*

g. *Requirements for security [RB, SSS; SRR];*

h. *Quality requirements [RB, SSS; SRR].*

ECSS-E-ST-40_0860717

b. Additional security requirements for software shall be allocated based on applicable security standards and regulations.

ECSS-E-ST-40_0860718

c. Where available, security requirements catalogues shall be considered in the allocation of security requirements to software, in agreement with the customer.

NOTE There are several pre-existing security requirements catalogues or controls that can be used (e.g., ISO 27002:2013, NIST SP 800-53, and ITSG-33).

ECSS-E-ST-40_0860719

d. The customer shall provide a traceability matrix to trace the security requirements to the results of the security analysis.

5.2.2.2 Identification of observability requirements

ECSS-E-ST-40_0860002

a. The customer shall specify all software observability requirements to monitor the software behaviour and to facilitate the system integration and failure investigation.

EXPECTED OUTPUT: System and software observability requirements [RB, SSS; SRR].

5.2.2.3 Specification of HMI requirements

ECSS-E-ST-40_0860003

a. The customer shall specify HMI requirements, following the human factor engineering process specified in ECSS-E-ST-10-11.

EXPECTED OUTPUT: HMI requirements [RB, SSS; SRR].

5.2.2.4 Hardware/Software co-engineering analysis

ECSS-E-ST-40_0860720

a. The customer shall perform a hardware/software co-engineering analysis to allocate requirements to software and hardware.

NOTE 1 The requirements allocated to software are specified in the SSS as identified in clause 5.2.2.1 and in IRD as identified in clause 5.2.4.3.

NOTE 2 The requirements allocated to hardware are specified at system level in accordance with ECSS-E-ST-20-40.

5.2.3 Software related system verification

5.2.3.1 Verification and validation process requirements

ECSS-E-ST-40_0860004

a. The customer shall specify the requirements needed for planning and setting up the system verification and validation process related to software.

EXPECTED OUTPUT: Verification and validation process requirements [RB, SSS; SRR].

ECSS-E-ST-40_0860721

b. Software verification and validation shall include requirements for planning and setting up system security verification and validation.

EXPECTED OUTPUT: Verification and validation process security requirements [RB, SSS; SRR].

5.2.3.2 System input for software validation

ECSS-E-ST-40_0860005

a. The customer shall specify requirements for the validation of the software against the requirements baseline and technical specification, in particular mission representative data and scenarios, and operational procedures to be used.

EXPECTED OUTPUT: Validation requirements and scenario [RB, SSS; SRR].

5.2.3.3 System input for software installation and acceptance

ECSS-E-ST-40_0860006

a. The customer shall specify requirements for the installation and acceptance of the software.

EXPECTED OUTPUT: Installation and acceptance requirements at the operational and maintenance sites [RB, SSS; SRR].

ECSS-E-ST-40_0860722

b. The customer shall specify requirements for secure installation and security specific acceptance criteria of the software.

EXPECTED OUTPUT: *Installation and acceptance requirements at the operational and maintenance sites [RB, SSS; SRR].*

5.2.4 Software related system integration and control

5.2.4.1 Identification of software versions for software integration into the system

ECSS-E-ST-40_0860007

a. The customer shall identify the software versions to be delivered and associate each requirement of the requirements baseline to a version.

EXPECTED OUTPUT: *Association of requirements to versions [RB, SSS; SRR].*

ECSS-E-ST-40_0860008

b. The customer shall specify the content and media of the delivery.

EXPECTED OUTPUT: *Delivery content and media [RB, SSS; SRR].*

5.2.4.2 Supplier support to system integration

ECSS-E-ST-40_0860009

a. The customer shall specify the support to be provided by the software supplier in order to integrate the software at system level.

EXPECTED OUTPUT: *System level integration support requirements [RB, SSS; SRR].*

NOTE For example: training, maintenance, configuration and test support.

5.2.4.3 Interface requirement specification

ECSS-E-ST-40_0860010

a. The customer shall specify the external interfaces of the software, including the static and dynamic aspects, for nominal and degraded modes.

EXPECTED OUTPUT: *External interface requirements specification [RB, IRD; SRR].*

NOTE External interfaces include interfaces with ASIC and FPGA.

ECSS-E-ST-40_0860723

b. The customer shall specify the security related aspects of external interfaces of the software.

EXPECTED OUTPUT: External interface requirements specification [RB, IRD; SRR].

5.2.4.4 System database

ECSS-E-ST-40_0860011

a. The customer shall specify the content of the system database for the supplier in order to ensure the consistency of common data and to define the allowed operational range of the data.

EXPECTED OUTPUT: System database content and allowed operational range [RB, SSS; SRR].

ECSS-E-ST-40_0860724

b. The customer shall specify the security constraints on the content of the system database as specified under 5.2.4.4a for the supplier, based on the security sensitivity of the data in the database.

5.2.4.5 Development constraints

ECSS-E-ST-40_0860012

a. The customer shall define specific development and design constraints on the supplier, including the use of development standards.

EXPECTED OUTPUT: Design and development constraints [RB, SSS; SRR].

NOTE This includes the identification of the information to be exchanged between Hardware and Software development teams (e.g. models, simulators, prototypes) and its planning.

5.2.4.6 On board control procedures

ECSS-E-ST-40_0860013

a. The customer shall specify the requirements to be implemented by OBCP.

EXPECTED OUTPUT: OBCP requirements [RB, SSS; SRR].

NOTE See ECSS-E-ST-70-01.

5.2.4.7 Development of software to be reused

ECSS-E-ST-40_0860014

a. The customer shall specify the reusability requirements that apply to the development, to enable the future reuse of the software (including models used to generate the software), or customization for mission.

EXPECTED OUTPUT: *Requirements for 'software to be reused' [RB, SSS; SRR].*

NOTE Customization is an important feature for family of spacecraft or launchers.

5.2.4.8 Software safety and dependability requirements

ECSS-E-ST-40_0860015

a. The customer shall specify the software safety and dependability requirements in accordance with ECSS-Q-ST-80 clauses 5.4.4, 6.2.2 and 6.2.3, based on the results of the safety and dependability analysis performed at system level.

EXPECTED OUTPUT: *Software safety and dependability requirements [RB, SSS; SRR].*

5.2.4.9 Format and data medium

ECSS-E-ST-40_0860016

a. The customer shall specify the format and the delivery medium of the exchanged data, in particular the interface and the system database.

EXPECTED OUTPUT: *Format and delivery medium of exchanged data [RB, SSS; SRR].*

ECSS-E-ST-40_0860725

b. The customer shall specify the security marking and labelling requirements of the exchanged data.

ECSS-E-ST-40_0860726

c. The customer shall specify the mechanisms to be used to ensure the integrity and authenticity of deliveries.

5.2.4.10 Security constraints for the development and integration environments

ECSS-E-ST-40_0860727

a. The customer shall define specific security constraints for the software development and integration environments, based on the security sensitivity of the software being developed.

EXPECTED OUTPUT: *Software development and integration security constraints [RB, SSS; SRR].*

5.2.4.11 Secure software delivery requirements

ECSS-E-ST-40_0860728

a. The customer shall specify requirements for secure software delivery.

EXPECTED OUTPUT: *Secure software delivery requirements [RB, SSS; SRR].*

NOTE 1 The aspect of corruption prevention of software delivery is covered by ECSS-Q-ST-80 6.2.4.8 and 6.2.4.9.

NOTE 2 The requirements in 5.2.4.9b apply to delivery media.

5.2.5 System requirements review

ECSS-E-ST-40_0860017

a. The customer shall conduct a system requirements review (SRR) in accordance with 5.3.4.1a.

5.3 Software management process

5.3.1 Overview

The management and control tasks specified in this clause are:

- software life cycle management;
- joint review process, software project reviews description, software technical reviews description, and reviews phasing;
- interface management;
- technical budget and margin management;
- compliance to this Standard.

5.3.2 Software life cycle management

5.3.2.1 Software life cycle identification

ECSS-E-ST-40_0860018

a. The software supplier shall define and follow a software life cycle including phases, their inputs and outputs, and joint reviews, in accordance with the overall project constraints and with ECSS-M-ST-10.

EXPECTED OUTPUT: *Software life cycle definition [MGT, SDP; SRR, PDR].*

ECSS-E-ST-40_0860019

b. The life cycle shall be chosen, assessing the specifics of the project technical approaches and the relevant project risks.

EXPECTED OUTPUT: Software life cycle definition [MGT, SDP; SRR, PDR].

ECSS-E-ST-40_0860020

c. The software supplier shall define the development strategy, the software engineering standards and techniques, the software development and the software testing environment.

EXPECTED OUTPUT: Development strategy, standards, techniques, development and testing environment [MGT, SDP; PDR].

ECSS-E-ST-40_0860021

d. The output of each phase and their status of completion, submitted as input to joint reviews, shall be specified in the software life cycle definition, including documents in complete or outline versions, and the results of verification of the outputs of the phase.

EXPECTED OUTPUT: Software life cycle definition [MGT, SDP; SRR, PDR].

5.3.2.2 Identification of interfaces between development and maintenance

ECSS-E-ST-40_0860022

a. The interfaces between development and maintenance shall be identified in the software life cycle.

EXPECTED OUTPUT: Identification of interface between development and maintenance [MGT, SDP; PDR].

NOTE This includes documents to be handed over and tools to be kept for maintenance.

5.3.2.3 Software procurement process implementation

ECSS-E-ST-40_0860023

a. The supplier shall document and implement the software procurement process as specified in ECSS-Q-ST-80 clause 5.5.

EXPECTED OUTPUT: Software procurement process documentation and implementation [MGT, SDP; SRR, PDR].

5.3.2.4 Automatic code generation

ECSS-E-ST-40_0860024

a. The autocode input models shall be reviewed together with the rest of the software specification, architecture and design.

EXPECTED OUTPUT: Autocode input model review [MGT, SDP; SRR, PDR].

NOTE The autocode input models are integral part of the software specification, architecture and design.

ECSS-E-ST-40_0860025

b. In the case of coexisting autocoded and manually written parts, the software development plan shall address the definition of a clear interface and resource allocation (memory and processing units).

EXPECTED OUTPUT: Autocode interface definition and resource allocation [MGT, SDP; SRR, PDR].

ECSS-E-ST-40_0860026

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].

ECSS-E-ST-40_0860027

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].

ECSS-E-ST-40_0860028

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].

5.3.2.5 Changes to baselines

ECSS-E-ST-40_0860029

a. Changes to baselines shall be handled by the configuration management process described in clause 6.2.4 of ECSS-Q-ST-80.

EXPECTED OUTPUT: Changes to baselines procedures [MGT, SCMP; SRR, PDR].

5.3.3 Joint review process

5.3.3.1 Joint reviews

ECSS-E-ST-40_0860030

a. Joint reviews shall be held to evaluate the progress and outputs of a project process or activity and provide evidence that:

1. the output are complete;
2. the output conforms to applicable standards and specifications;
3. any changes are properly implemented and impact only those areas identified by the configuration management process;
4. the output conforms to applicable schedules;
5. the output are in such a status that the next activity can start;
6. the activity is being conducted according to the plans, schedules, standards, and guidelines laid down for the project;
7. any security measures defined on the basis of the security analysis are being correctly implemented.

EXPECTED OUTPUT: Joint review reports [DJF, - ; SRR, PDR, CDR, QR, AR].

NOTE The joint review process is a process for evaluating the status and products of an activity of a project as appropriate. This process is employed by two parties, where one party (reviewing party) reviews another party (reviewed party). For project reviews, the two parties are the customer and the supplier. Joint reviews are held throughout the life cycle of the software;

ECSS-E-ST-40_0860729

b. The planning and execution of joint reviews shall conform with the measures and regulations imposed by the security sensitivity of the input and outputs of the review.

NOTE Applicable measures and regulations can imply organising security specific sessions.

5.3.3.2 Software project reviews

ECSS-E-ST-40_0860031

a. Software project reviews, i.e. joint reviews organized under the responsibility of the customer aiming at defining a customer approved technical baseline shall be included in the software life cycle, with as a minimum SRR, PDR, CDR, QR and AR as specified in 5.3.4.

EXPECTED OUTPUT: Software project reviews included in the software life cycle definition [MGT, SDP; SRR, PDR].

ECSS-E-ST-40_0860032

b. The review process specified in ECSS-M-ST-10-01 shall apply to all software project reviews, including the agreement on a review plan before the review process is started.

EXPECTED OUTPUT: *Review Plan [MGT, SRevP; SRR, PDR].*

5.3.3.3 Software technical reviews

ECSS-E-ST-40_0860033

a. In addition to the software project reviews, software technical reviews, i.e. joint reviews organized under the responsibility of the customer or the supplier aiming at defining a technical baseline, shall be defined.

EXPECTED OUTPUT: *Software technical reviews included in the software life cycle definition [MGT, SDP; SRR, PDR].*

ECSS-E-ST-40_0860034

b. The applicable technical review process shall be specified by the supplier.

EXPECTED OUTPUT: *Technical reviews process [MGT; SDP; SRR, PDR].*

ECSS-E-ST-40_0860035

c. The supplier shall use the software technical reviews to implement intermediate reviews.

EXPECTED OUTPUT: *Software technical reviews included in the software life cycle definition [MGT, SDP; SRR, PDR].*

5.3.4 Software project reviews description

5.3.4.1 System requirement review

ECSS-E-ST-40_0860036

a. After completion of the software requirements baseline specification, a system requirements review (SRR) shall take place.

AIM: Reach the approval of the software requirements baseline by all stakeholders.

EXPECTED OUTPUT: *Approved requirements baseline [RB; SRR].*

5.3.4.2 Preliminary design review

ECSS-E-ST-40_0860037

a. After completion of the software requirement analysis and architectural design, and the verification and validation processes implementation, a preliminary design review (PDR) shall take place.

AIM: To review compliance of the technical specification (TS) with the requirements baseline, to review the software architecture and interfaces, to review the development, verification and validation plans.

EXPECTED OUTPUT: *Approved technical specification and interface, architecture and plans [TS, DDF, DJF, MGT; PDR].*

ECSS-E-ST-40_0860038

b. In case the software requirements are baselined before the start of the architectural design, the part of the PDR addressing the software requirements specification and the interfaces specification shall be held in a separate joint review anticipating the PDR, in a software requirements review (SWRR).

AIM: To baseline the software requirements before the start of the architectural design.

EXPECTED OUTPUT: *Approved technical specification and interface [TS; PDR].*

NOTE e.g. in case of software intensive system or when an early baseline of the requirements is required.

5.3.4.3 Critical design review

ECSS-E-ST-40_0860039

a. After completion of the design of software items, coding and testing, integration and validation with respect to the technical specification, a critical design review (CDR) shall take place.

AIM: —To review the design definition file, including software architectural design, detailed design, code and user manuals;

— To review the design justification file, including the completeness of the software unit testing, integration and validation with respect to the technical specification.

EXPECTED OUTPUT: *Approved design definition file and design justification file [DDF, DJF; CDR].*

ECSS-E-ST-40_0860040

b. In case the software detailed design is baselined before the start of the coding, the part of the CDR addressing the software detailed design, the interfaces design and the software budget shall be held in a separate joint review anticipating the CDR, in a detailed design review (DDR).

AIM: To baseline the software detailed design before the start of the coding.

EXPECTED OUTPUT: *Approved detailed design, interface design and budget [DDF, DJF; CDR].*

NOTE e.g. in case of complex software products or when an early baseline of the design is required.

5.3.4.4 Qualification review

ECSS-E-ST-40_0860041

a. After completion of the software validation against the requirements baseline, and the verification activities, a qualification review (QR) shall take place.

AIM: To ensure that verification and validation processes have been completed successfully.

EXPECTED OUTPUT: Qualified software product [RB, TS, DDF, DJF, MGT, MF; QR].

5.3.4.5 Acceptance review

ECSS-E-ST-40_0860042

a. After completion of the software delivery and installation, and software acceptance, an acceptance review (AR) shall take place.

AIM: To accept the software product in the intended operational environment.

EXPECTED OUTPUT: Accepted software product [RB, TS, DDF, DJF, MGT, MF; AR].

5.3.5 Software technical reviews description

5.3.5.1 Test readiness reviews

ECSS-E-ST-40_0860043

a. Test readiness reviews (TRR) shall be held before the beginning of test activities, as defined in the software development plan.

AIM: To ensure that the software status is compatible with the commencement of test activities and the necessary resources are available and ready for use.

EXPECTED OUTPUT: Confirmation of readiness of test activities [DJF; TRR].

5.3.5.2 Test review board

ECSS-E-ST-40_0860044

a. The test review board (TRB) shall convene at the end of test activities, as defined in the software development plan to verify the test results with respect to the testing specification and plans.

EXPECTED OUTPUT: Approved test results[DJF; TRB].

5.3.5.3 Software delivery review board

ECSS-E-ST-40_0860730

a. A software delivery review board (SW DRB) shall convene before the delivery of any software version in order to assess if the delivery is suitable for the intended use.

EXPECTED OUTPUT: Acceptance of the software delivery [DJF; SW DRB].

5.3.5.4 Software validation specification review

ECSS-E-ST-40_0860731

a. The software validation specification review (SVSR) shall be held before or combined with the TRR.

AIM: To ensure that the software validation specification is complete and gives confidence that the requirements will be properly validated.

EXPECTED OUTPUT: *Approved software validation specification [DJF, SVS; SVSR]*

5.3.6 Review phasing

5.3.6.1 Review phasing for flight software

ECSS-E-ST-40_0860045

a. For flight software, the phasing of the software life cycle to the system life cycle shall be chosen, assessing the following driving aspects:

1. the system model philosophy,
2. the system verification and qualification approach and constraints,
3. the capability to baseline the system design at system CDR, by knowing enough information about software design, in particular consolidated sizing and timing budgets, consistent hardware design and software design.

EXPECTED OUTPUT: *Flight software review phasing [MGT, SDP; SRR, PDR].*

NOTE An example of system model philosophy is the use of protoflight model versus the utilization of engineering qualification model.

ECSS-E-ST-40_0860046

b. For flight software, the following software versus system level reviews synchronisation shall be planned as follows:

1. the software SRR not later than the system PDR,
2. the software PDR between the system PDR and the system CDR,
3. the detailed design of the software reviewed before the system CDR,
4. the software CDR before the system QR,
5. the software QR within system QR.

EXPECTED OUTPUT: *Flight software review phasing [MGT, SDP; SRR, PDR]*

NOTE 1 In case multiple reviews of a certain kind (SRR, PDR, CDR) are foreseen by the chosen lifecycle, the requirement refers to the first occurrence of the review.

NOTE 2 The detailed design of the software can be reviewed during a DDR in anticipation to software CDR.

5.3.6.2 Review phasing for ground software

ECSS-E-ST-40_0860047

a. For ground segment software, the software life cycle shall be chosen assessing the following constraints for the ground reviews phasing:

1. the review of software requirements is performed in a SWRR or PDR before the ground segment PDR,
2. the software PDR is performed before the ground segment CDR,
3. all the other software reviews are performed before the ground segment QR.

EXPECTED OUTPUT: *Ground software review phasing [MGT, SDP; SRR, PDR]*

NOTE In case multiple reviews of a certain kind (SWRR, PDR, CDR) are foreseen by the chosen lifecycle, the requirement refers to the first occurrence of the review.

5.3.7 Interface management

5.3.7.1 Interface management procedures

ECSS-E-ST-40_0860048

a. Interface management procedures shall be defined in accordance with ECSS-M-ST-40 requirements.

AIM: Define procedures that guarantee the consistency of the system interfaces.

EXPECTED OUTPUT: *Interface management procedures [MGT, - ; SRR].*

5.3.8 Technical budget and margin management

5.3.8.1 Software technical budget and margin philosophy definition

ECSS-E-ST-40_0860049

a. Technical budget targets and margin philosophy dedicated to the software shall be specified by the customer in the requirements baseline in order to define the limits of software budgets associated with computer and network resources such as PU load, maximum memory size, deadline fulfilment, communication, archiving needs, remote access needs and performance requirements such as data throughput.

AIM: This allows anticipating:

- Expected changes in the requirements baseline,
- Requirements on reprogramming of the system during operational use,
- Required budget for temporary copies of software images,
- Constraints on state transitions, especially when recovery from a faulty state is concerned,
- Constraints on physical processor (CPU) type and memory and expected processor capacity, wait states, interfaces, caching and pipelining, interference scheme between processing units, etc.,
- Equipment, communication and performances aspects such as buses, protocols, acceptable errors, bus capacity usage by other sources.,
- Accuracy aspects, such as conversion to/from analogue signals, and accuracy of timing signals,
- Budgets for OS kernel characterisation, such as context switch latency or deadlines for tasking,
- Mission and system operation characteristics and reference operational scenarios.

EXPECTED OUTPUT: Technical budgets and margin philosophy for the project [RB, SSS; SRR].

5.3.8.2 Technical budget and margin computation

ECSS-E-ST-40_0860050

a. For PU load, the way to compute the technical budgets and margin shall be agreed between the customer and the supplier.

EXPECTED OUTPUT: Technical budgets and margin computation [DJF, SVR; SRR, PDR]

NOTE 1 Margins for PU load can be proposed:

- per task in isolation, by provision of a margin of execution time (margin_WCET);
- per task in context, by provision of a growth capacity before it reaches its deadline (margin_slack);
- globally, by provision of a growth capacity before the PU saturation (margin utilisation).

NOTE 2 The PU load margins can be based on either theoretical worst case or operational worst case.

ECSS-E-ST-40_0860732

b. For memory, the margin shall be defined as follows:
margin_memory = free size / total size

EXPECTED OUTPUT: Technical budgets and margin computation [DJF, SVR; SRR, PDR].

5.3.9 Compliance to this Standard

5.3.9.1 Compliance matrix

ECSS-E-ST-40_0860051

a. The supplier shall provide a compliance matrix documenting conformance with the individual software engineering process requirements (Clause 5) applicable to the project or business agreement, as per ECSS-S-ST-00.

EXPECTED OUTPUT: *ECSS-E-ST-40 compliance matrix [MGT, SDP; SRR, PDR].*

5.3.9.2 Documentation compliance

ECSS-E-ST-40_0860052

a. The compliance to each of the individual software engineering process requirements shall make reference to the document where the expected output is placed, if it is not placed in this Standard's DRDs in annexes of this document.

EXPECTED OUTPUT: *ECSS-E-ST-40 compliance matrix [MGT, SDP; SRR, PDR].*

5.4 Software requirements and architecture engineering process

5.4.1 Overview

The software requirements and architecture engineering process consists of the following activities:

- software requirements analysis;
- software architectural design.

5.4.2 Software requirements analysis

5.4.2.1 Establishment and documentation of software requirements

ECSS-E-ST-40_0860053

a. The supplier shall establish and document software requirements, including the software quality requirements, as part of the technical specification.

EXPECTED OUTPUT: *The following outputs are expected:*

a. Functional and performance specifications, including hardware characteristics, and environmental conditions under which the *software item executes, including budgets requirements [TS, SRS; PDR];*

b. *Operational, reliability, safety, maintainability, portability, configuration, delivery, adaptation and installation requirements, design constraints [TS, SRS; PDR];*

c. *Software product quality requirements (see ECSS-Q-ST-80 clause 7.2) [TS, SRS; PDR];*

d. *Security specifications, including those related to factors which can compromise sensitive information [TS, SRS, ICD; PDR];*

e. *Human factors engineering (ergonomics including HMI usability) specifications, following the human factor engineering process specified in ECSS-E-ST-10-11 [TS, SRS; PDR];*

f. *Data definition and database requirements [TS, SRS; PDR];*

g. *Validation requirements [TS, SRS, ICD; PDR]*

h. *Interfaces external to the software item [TS, ICD; PDR];*

i. *Reuse requirements (see ECSS-Q-ST-80 clause 6.2.7) [TS, SRS; PDR];*

j. *Requirements to contribute to the treatment of security risks that are evaluated not acceptable [TS, SRS, ICD; PDR].*

NOTE Protection of private data is part of the security specifications.

5.4.2.2 Definition of functional and performance requirements for in flight modification

ECSS-E-ST-40_0860054

a. When in flight modification is specified for flight software, the supplier shall perform analysis of the specific implications for the software design and validation processes and include the functional and performance requirements in the technical specification, including in case of use of automatic code generation.

EXPECTED OUTPUT: Specifications for in flight software modifications [TS, SRS; PDR].

5.4.2.3 Construction of a software logical model

ECSS-E-ST-40_0860055

a. The supplier shall construct a logical model of the functional requirements of the software product.

EXPECTED OUTPUT: Software logical model [TS, SRS; PDR].

ECSS-E-ST-40_0860056

b. The supplier shall use a method to support the construction of the logical model.

EXPECTED OUTPUT: *Software logical model method [TS, SRS; PDR].*

ECSS-E-ST-40_0860057

c. The logical model shall include a behavioural view.

EXPECTED OUTPUT: *Behavioural view in software logical model [TS, SRS; PDR].*

NOTE The software logical model makes the software requirements understandable as a whole and not just individually. Depending on the notation, modelling language or method that is used to express it, it can also be used for:

- verification of completeness,
- verification of consistency,
- proof or formal verification of some high-level properties,
- generation of test scenarios.

The model shows the software functionalities and include behavioural views.

5.4.2.4 Conducting a software requirement review

ECSS-E-ST-40_0860058

a. The supplier shall conduct a software requirement review (SWRR) as anticipation of the PDR, in conformance with 5.3.4.2b.

5.4.3 Software architectural design

5.4.3.1 Transformation of software requirements into a software architecture

ECSS-E-ST-40_0860059

a. The supplier shall transform the requirements for the software item into an architecture that:

1. describes its top–level structure;
2. identifies the software components, ensuring that all the requirements for the software item are allocated to its software components and later refined to facilitate detailed design;
3. covers as a minimum hierarchy, dependency, interfaces and operational usage for the software components;
4. documents the process, data and control aspects of the product;

5. describes the architecture static decomposition into software elements such as packages, classes or units;

6. describes the dynamic architecture, which involves the identification of active objects such as threads, tasks and processes;

7. describes the software behaviour.

EXPECTED OUTPUT: *Software architectural design [DDF, SDD; PDR].*

5.4.3.2 Software design method

ECSS-E-ST-40_0860060

a. The supplier shall use a method to produce the static and dynamic architecture including:

1. software elements, their interfaces;

2. software elements relationships.

EXPECTED OUTPUT: *Software architectural design method [DDF, SDD; PDR].*

NOTE The method can be object oriented or functional.

ECSS-E-ST-40_0860733

b. The supplier shall perform a residual vulnerability analysis of the chosen architectural design.

EXPECTED OUTPUT: *List of vulnerabilities [SF, -; PDR].*

5.4.3.3 Selection of a computational model for real–time software

ECSS-E-ST-40_0860061

a. The dynamic architecture design shall be described according to an analysable computational model.

EXPECTED OUTPUT: *Computational model [DDF, SDD; PDR].*

5.4.3.4 Description of software behaviour

ECSS-E-ST-40_0860062

a. The software architecture design shall also describe the behaviour of the software, by means of description techniques using automata and scenarios.

EXPECTED OUTPUT: *Software behaviour [DDF, SDD; PDR].*

5.4.3.5 Development and documentation of the software interfaces

ECSS-E-ST-40_0860063

a. The supplier shall develop and document a software preliminary design for the interfaces external to the software item and between the software components of the software item.

EXPECTED OUTPUT: *The following outputs are expected:*

a. Preliminary external interfaces design [TS, ICD; PDR];

b. Preliminary internal interfaces design [DDF, SDD; PDR].

5.4.3.6 Definition of methods and tools for software intended for reuse

ECSS-E-ST-40_0860064

a. The supplier shall define procedures, methods and tools for reuse, and apply these to the software engineering processes to comply with the reusability requirements for the software development.

EXPECTED OUTPUT: *Software intended for reuse - justification of methods and tools [DJF, SRF; PDR].*

ECSS-E-ST-40_0860065

b. An evaluation of the reuse potential of the software shall be performed at PDR and CDR.

EXPECTED OUTPUT: *Software intended for reuse - evaluation of reuse potential [DJF, SRF; PDR, CDR].*

ECSS-E-ST-40_0860066

c. The supplier shall design the software such that mission and configuration dependant data are separated.

EXPECTED OUTPUT: *Software architectural design with configuration data - [DDF, SDD; PDR, CDR].*

NOTE It is a common practice to use database to store the mission and configuration dependant data.

5.4.3.7 Reuse of existing software

ECSS-E-ST-40_0860067

a. The analysis of the potential reusability of existing software components shall be performed through:

1. identification of the reuse components and models with respect to the requirements baseline;

2. a quality evaluation of these components, applying ECSS-Q-ST-80 clause 6.2.7.

EXPECTED OUTPUT: *Justification of reuse with respect to requirements baseline [DJF, SRF; PDR].*

5.4.3.8 Definition and documentation of the software integration requirements and plan

ECSS-E-ST-40_0860068

a. The supplier shall define and document the preliminary software integration strategy in terms of responsibility and schedule, control procedures and testing approach identifying the goals to be achieved, sequence, environment and criteria.

EXPECTED OUTPUT: *Software integration strategy [DJF, SUITP; PDR].*

5.4.4 Conducting a preliminary design review

ECSS-E-ST-40_0860069

a. The supplier shall conduct a preliminary design review (PDR) in accordance with clause 5.3.4.2.

NOTE The successful completion of the review establishes a baseline for the development of the software item.

5.5 Software design and implementation engineering process

5.5.1 Overview

The software design and implementation engineering process consists of the following activities:

- design of software items;
- coding and testing;
- integration.

5.5.2 Design of software items

5.5.2.1 Detailed design of each software component

ECSS-E-ST-40_0860070

a. The supplier shall develop a detailed design for each component of the software and document it.

EXPECTED OUTPUT: *Software components design documents [DDF, SDD; CDR].*

ECSS-E-ST-40_0860071

b. Each software component shall be refined into lower levels containing software units that can be coded, compiled, and tested.

EXPECTED OUTPUT: *Software components design documents [DDF, SDD; CDR].*

ECSS-E-ST-40_0860072

c. It shall be ensured that all the software requirements are allocated from the software components to software units.

EXPECTED OUTPUT: *Software components design documents [DDF, SDD; CDR].*

5.5.2.2 Development and documentation of the software interfaces detailed design

ECSS-E-ST-40_0860073

a. The supplier shall develop and document a detailed design for the interfaces external to the software item, between the software components, and between the software units, in order to allow coding without requiring further information.

EXPECTED OUTPUT: *The following outputs are expected:*

a. External interfaces design (update) [TS, ICD; CDR];

b. Internal interfaces design (update) [DDF, SDD; CDR].

5.5.2.3 Production of the detailed design model

ECSS-E-ST-40_0860074

a. The supplier shall produce the detailed design model of the software components defined during the software architectural design, including their static, dynamic and behavioural aspects.

EXPECTED OUTPUT: *The following outputs are expected:*

a. Software static design model [DDF, SDD; CDR];

b. Software dynamic design model [DDF, SDD; CDR];

c. Software behavioural design model [DDF, SDD; CDR].

5.5.2.4 Software detail design method

ECSS-E-ST-40_0860075

a. The supplier shall use a design method to produce the detailed design including:

1. software units, their interfaces;
2. software units relationships.

EXPECTED OUTPUT: *Software design method [DDF, SDD; CDR].*

NOTE The method can be object oriented or functional.

5.5.2.5 Detailed design of real–time software

ECSS-E-ST-40_0860076

a. The dynamic design model shall be compatible with the computational model selected during the software architectural design model.

EXPECTED OUTPUT: *Real-time software dynamic design model [DDF, SDD; CDR].*

ECSS-E-ST-40_0860077

b. The supplier shall document and justify all timing and synchronization mechanisms.

EXPECTED OUTPUT: *Real-time software dynamic design model [DDF, SDD; CDR].*

ECSS-E-ST-40_0860078

c. The supplier shall document and justify all the design mutual exclusion mechanisms to manage access to the shared resources.

EXPECTED OUTPUT: *Real-time software dynamic design model [DDF, SDD; CDR].*

ECSS-E-ST-40_0860079

d. The supplier shall document and justify the use of dynamic allocation of resources.

EXPECTED OUTPUT: *Real-time software dynamic design model [DDF, SDD; CDR].*

ECSS-E-ST-40_0860080

e. The supplier shall ensure protection against problems that can be induced by the use of dynamic allocation of resources.

EXPECTED OUTPUT: *Real-time software dynamic design model [DDF, SDD; CDR].*

5.5.2.6 Utilization of description techniques for the software behaviour

ECSS-E-ST-40_0860081

a. The behavioural design of the software units shall be described by means of techniques using automata and scenarios.

EXPECTED OUTPUT: *Software behavioural design model techniques [DDF, SDD; CDR].*

5.5.2.7 Determination of design method consistency for real–time software

ECSS-E-ST-40_0860082

a. It shall be ensured that all the methods utilized for different item of the same software are, from a dynamic stand–point, consistent among themselves and consistent with the selected computational model.

EXPECTED OUTPUT: Compatibility of real-time design methods with the computational model [DDF, SDD; CDR].

5.5.2.8 Development and documentation of the software user manual

ECSS-E-ST-40_0860083

a. The supplier shall develop and document the software user manual.

EXPECTED OUTPUT: Software user manual [DDF, SUM; CDR].

5.5.2.9 Definition and documentation of the software unit test requirements and plan

ECSS-E-ST-40_0860084

a. The supplier shall define and document responsibility and schedule, control procedures, testing approach, test design and test case specification for testing software units.

EXPECTED OUTPUT: Software unit test plan [DJF, SUITP; CDR].

5.5.2.10 Conducting a detailed design review

ECSS-E-ST-40_0860085

a. The supplier shall conduct a detailed design review (DDR) as anticipation of the CDR, in conformance with 5.3.4.3b.

5.5.3 Coding and testing

5.5.3.1 Development and documentation of the software units

ECSS-E-ST-40_0860086

a. The supplier shall develop and document the following:

1. the coding of each software unit;
2. the build procedures to compile and link software units;

EXPECTED OUTPUT: The following outputs are expected:

a. Software component design documents and code (update) [DDF, SDD, source code; CDR];

b. Software configuration file - build procedures [DDF, SCF; CDR].

5.5.3.2 Software unit testing

ECSS-E-ST-40_0860087

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].

ECSS-E-ST-40_0860088

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].

ECSS-E-ST-40_0860089

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 document;
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].

5.5.4 Integration

5.5.4.1 Software integration test plan development

ECSS-E-ST-40_0860090

a. The supplier shall complement the software integration test plan to define the integration of the software units and software components into the software item, providing the following data:

1. test design;
2. test case specification;
3. test procedures;
4. test data.

EXPECTED OUTPUT: Software integration test plan (update) [DJF, SUITP; CDR].

5.5.4.2 Software units and software component integration and testing

ECSS-E-ST-40_0860091

a. The supplier shall integrate the software units and software components, and test them, as the aggregates are developed, in accordance with the integration plan, ensuring that each aggregate satisfies the requirements of the software item and that the software item is integrated at the conclusion of the integration activity.

EXPECTED OUTPUT: Software integration test report [DJF, - ; CDR].

5.6 Software validation process

5.6.1 Overview

The software validation process consists of:

- validation process implementation,
- validation activities with respect to the technical specification, and
- validation activities with respect to the requirements baseline.

5.6.2 Validation process implementation

5.6.2.1 Establishment of a software validation process

ECSS-E-ST-40_0860092

a. The validation process shall be established to validate the software product.

EXPECTED OUTPUT: Software validation plan - validation process identification [DJF, SValP; PDR].

ECSS-E-ST-40_0860093

b. Validation tasks defined in clauses 5.6.3 and 5.6.4 including associated methods, techniques, and tools for performing the tasks, shall be selected and the regression test strategy specified.

EXPECTED OUTPUT: Software validation plan - methods and tools [DJF, SValP; PDR].

ECSS-E-ST-40_0860094

c. The validation effort and the degree of organizational independence of that effort shall be determined, coherent with ECSS-Q-ST-80 clause 6.3.5.19.

EXPECTED OUTPUT: Software validation plan - effort and independence [DJF, SValP; PDR].

ECSS-E-ST-40_0860734

d. The validation activities shall comply with applicable organisational regulations and policies.

5.6.2.2 Selection of the organization responsible for conducting the independent validation

ECSS-E-ST-40_0860095

a. If the project warrants an independent validation effort, a qualified organization responsible for conducting the effort shall be selected.

EXPECTED OUTPUT: *Independent software validation plan - organization selection [DJF, - ; PDR].*

ECSS-E-ST-40_0860096

b. The customer shall ensure the independence and authority of the organization selected to perform the independent validation tasks.

EXPECTED OUTPUT: *Independent software validation plan - level of independence [DJF, - ; PDR].*

NOTE See also ECSS-Q-ST-80, clause 6.3.5.28 (independent software validation).

5.6.3 Validation activities with respect to the technical specification

5.6.3.1 Development and documentation of a software validation specification with respect to the technical specification

ECSS-E-ST-40_0860097

a. The supplier shall develop and document, for each requirement of the software item in TS (including ICD), a set of tests, test cases (inputs, outputs, test criteria) and test procedures including:

1. testing with stress, boundary, and singular inputs;
2. testing the software product for its ability to isolate and reduce the effect of errors;
3. testing that the software product can perform successfully in a representative operational environment;
4. external interface testing including boundaries, protocols and timing test;
5. testing HMI applications as per ECSS-E-ST-10-11;
6. security testing.

EXPECTED OUTPUT: *Software validation specification with respect to the technical specification [DJF, SVS; CDR].*

NOTE 1 A test case identifies inputs, outputs, and test criteria.

NOTE 2 The reduction of the effect of an error can be done by graceful degradation upon failure, request for operator assistance upon stress, boundary and singular conditions.

ECSS-E-ST-40_0860098

b. Validation shall be performed by test.

EXPECTED OUTPUT: Software validation specification with respect to the technical specification [DJF, SVS; CDR].

ECSS-E-ST-40_0860099

c. If it can be justified that validation by test cannot be performed, validation shall be performed by either analysis, inspection or review of design.

EXPECTED OUTPUT: Software validation specification with respect to the technical specification [DJF, SVS; CDR].

5.6.3.2 Conducting the validation with respect to the technical specification

ECSS-E-ST-40_0860100

a. The validation shall be conducted as specified in the output of clause 5.6.3.1.

EXPECTED OUTPUT: Software validation report with respect to the technical specification [DJF, - ; CDR].

5.6.3.3 Updating the software user manual

ECSS-E-ST-40_0860101

a. The supplier shall update the software user manual in accordance with the results of the validation activities with respect to the technical specification.

EXPECTED OUTPUT: Software user manual (update) [DDF, SUM; CDR].

5.6.3.4 Conducting a critical design review

ECSS-E-ST-40_0860102

a. The supplier shall conduct a critical design review (CDR) in accordance with clause 5.3.4.3.

5.6.4 Validation activities with respect to the requirements baseline

5.6.4.1 Development and documentation of a software validation specification with respect to the requirements baseline

ECSS-E-ST-40_0860103

a. The supplier shall develop and document, for each requirement of the software item in RB, including IRD, a set of tests, test cases and test procedures including:

1. testing against the mission data and scenario specified by the customer in 5.2.3.1;
2. testing with stress, boundary, and singular inputs;
3. testing the software product for its ability to isolate and reduce the effect of errors;
4. testing that the software product can perform successfully in a representative operational and non-intrusive environment;
5. external interface testing including boundaries, protocols and timing test;
6. testing HMI applications as per ECSS-E-ST-10-11;
7. security testing against security requirements as per clause 5.2.2.1.

EXPECTED OUTPUT: *Software validation specification with respect to the requirements baseline [DJF, SVS; QR, AR].*

NOTE 1 A test case identifies inputs, outputs, and test criteria.

NOTE 2 The reduction of the effect of an error can be done by graceful degradation upon failure, request for operator assistance upon stress, boundary and singular conditions.

ECSS-E-ST-40_0860104

b. Validation shall be performed by test.

EXPECTED OUTPUT: *Software validation specification with respect to the requirements baseline [DJF, SVS; QR, AR].*

ECSS-E-ST-40_0860105

c. If it can be justified that validation by test cannot be performed, validation shall be performed by either analysis, inspection or review of design.

EXPECTED OUTPUT: *Software validation specification with respect to the requirements baseline [DJF, SVS; QR, AR].*

5.6.4.2 Conducting the validation with respect to the requirements baseline

ECSS-E-ST-40_0860106

a. The validation shall be conducted as specified in the output of clause 5.6.4.1.

EXPECTED OUTPUT: *Software validation report with respect to the requirements baseline [DJF, - ; QR, AR].*

ECSS-E-ST-40_0860107

b. The validation tests shall be “black box”, i.e. performed on the final software product to be delivered, without any modification of the code or of the data.

EXPECTED OUTPUT: *Software validation report with respect to the requirements baseline [DJF, - ; QR, AR].*

NOTE In particular, this is essential when a mission database is used to customize the final product, and when late versions of the database are used to update the software.

5.6.4.3 Updating the software user manual

ECSS-E-ST-40_0860108

a. The supplier shall update the software user manual in accordance with the results of the validation activities with respect to the requirements baseline.

EXPECTED OUTPUT: *Software user manual (update) [DDF, SUM; QR, AR].*

5.6.4.4 Conducting a qualification review

ECSS-E-ST-40_0860109

a. The qualification review (QR) shall be conducted in accordance with clause 5.3.4.4.

5.6.5 Software validation control

ECSS-E-ST-40_0860735

a. The supplier shall maintain up-to-date and provide to the customer software validation control information, containing at least the following information for each RB and TS requirement:

1. Validation execution: executed, partially executed, not executed;
2. Validation result: successful, failed, pending;
3. References to the test cases validating the requirement;
4. The configuration identification of the test environment used for validation and of the software being validated;

5. The date, time and responsible personnel for the validation;
6. References to the specific validation logs;
7. References to any Software Problem Reports (SPR's) raised as a result of the validation.

NOTE The information is expected to be provided in an electronic format to be agreed with the customer.

5.7 Software delivery and acceptance process

5.7.1 Overview

This process consists of the following activities:

- software delivery and installation;
- software acceptance.

5.7.2 Software delivery and installation

5.7.2.1 Preparation of the software product

ECSS-E-ST-40_0860110

a. The supplier shall prepare the deliverable software product for its installation in the target platform.

EXPECTED OUTPUT: The following outputs are expected:

a. Software product [DDF, - ; QR, AR];

b. Software release document [DDF, SRelD; QR, AR].

ECSS-E-ST-40_0860736

b. The supplier shall take measures to ensure secure software delivery.

5.7.2.2 Supplier's provision of training and support

ECSS-E-ST-40_0860111

a. The supplier shall provide initial and continuing training and support to the customer if specified in the requirements baseline.

EXPECTED OUTPUT: Training material [DDF, - ; QR].

5.7.2.3 Installation procedures

ECSS-E-ST-40_0860112

a. The supplier shall develop procedures to install the software product in the target environment.

EXPECTED OUTPUT: Installation procedures [DDF, SCF ; AR].

ECSS-E-ST-40_0860737

b. The installation procedures shall ensure secure installation of the software.

5.7.2.4 Installation activities reporting

ECSS-E-ST-40_0860113

a. The resources and information to install the software product shall be determined and be available.

EXPECTED OUTPUT: Installation report [DJF, - ; AR].

ECSS-E-ST-40_0860114

b. The supplier shall assist the customer with the set–up activities.

EXPECTED OUTPUT: Installation report [DJF, - ; AR].

ECSS-E-ST-40_0860115

c. It shall be ensured that the software code and databases initialize, execute and terminate as specified in the installation plan.

EXPECTED OUTPUT: Installation report [DJF, - ; AR].

ECSS-E-ST-40_0860116

d. The installation events and results shall be documented.

EXPECTED OUTPUT: Installation report [DJF, - ; AR].

5.7.3 Software acceptance

5.7.3.1 Acceptance test planning

ECSS-E-ST-40_0860117

a. The customer shall establish an acceptance test plan specifying the intended acceptance tests with tests suited to the target environment.

EXPECTED OUTPUT: Acceptance test plan [DJF, - ; QR, AR].

5.7.3.2 Acceptance test execution

ECSS-E-ST-40_0860118

a. The customer shall perform the acceptance testing.

EXPECTED OUTPUT: Acceptance test report [DJF, - ; AR].

5.7.3.3 Executable code generation and installation

ECSS-E-ST-40_0860119

a. The acceptance shall include generation of the executable code from configuration managed source code components and its installation on the target environment.

EXPECTED OUTPUT: *Software product [DDF, - ; AR].*

5.7.3.4 Supplier’s support to customer’s acceptance

ECSS-E-ST-40_0860120

a. The supplier shall support the customer’s acceptance reviews and testing of the software product in preparation of the AR.

EXPECTED OUTPUT: *Joint review reports [DJF, -; AR].*

NOTE Acceptance reviews and testing considers the results of the joint reviews (see 5.3.3), audits, testing and validation (see ECSS-Q-ST-80 clauses 5.2.3 and 6.3.5), and system validation testing (if performed).

ECSS-E-ST-40_0860121

b. The results of the acceptance reviews and testing shall be documented.

EXPECTED OUTPUT: *Joint review reports [DJF, -; AR].*

5.7.3.5 Evaluation of acceptance testing

ECSS-E-ST-40_0860122

a. The acceptance tests shall be traced to the requirements baseline.

EXPECTED OUTPUT: *Traceability of acceptance tests to the requirements baseline [DJF, SVR; AR].*

5.7.3.6 Conducting an acceptance review

ECSS-E-ST-40_0860123

a. The acceptance review (AR) shall be conducted in accordance with clause 5.3.4.5.

5.8 Software verification process

5.8.1 Overview

The software verification process consists of:

- verification process implementation, and
- verification activities.

5.8.2 Verification process implementation

5.8.2.1 Establishment of a software verification process

ECSS-E-ST-40_0860124

a. The verification process shall be established by the supplier to verify the software products.

EXPECTED OUTPUT: *Software verification plan - verification process identification [DJF, SVerP; PDR].*

ECSS-E-ST-40_0860125

b. Life cycle activities and software products needing verification shall be determined based upon the scope, magnitude, complexity, and criticality analysis.

EXPECTED OUTPUT: *Software verification plan - software products identification [DJF, SVerP; PDR].*

ECSS-E-ST-40_0860126

c. Verification activities and tasks defined in clause 5.8.3, including associated methods, techniques, and tools for performing the tasks, shall be selected for the life cycle activities and software products.

EXPECTED OUTPUT: *Software verification plan - activities, methods and tools [DJF, SVerP; PDR].*

ECSS-E-ST-40_0860127

d. A determination shall be made concerning the verification effort, the identification of risks and the degree of organizational independence.

EXPECTED OUTPUT: *Software verification plan - organizational independence, risk and effort identification [DJF, SVerP; PDR].*

ECSS-E-ST-40_0860738

e. The verification activities shall conform with applicable security organisational regulations and policies.

5.8.2.2 Selection of the organization responsible for conducting the independent verification

ECSS-E-ST-40_0860128

a. If the project warrants an independent verification effort, a qualified organization shall be selected for conducting the verification.

EXPECTED OUTPUT: *Independent software verification plan - organization selection [DJF, - ; PDR].*

ECSS-E-ST-40_0860129

b. The customer shall ensure the independence and authority of the organization selected to perform the independent verification activities.

AIM: A coherent and consistent approach to project organization within each project.

EXPECTED OUTPUT: *Independent software verification plan - level of independence [DJF, - ; PDR].*

NOTE See also ECSS-Q-ST-80 clause 6.2.6.13 (independent software verification).

5.8.3 Verification activities

5.8.3.1 Verification of requirements baseline

ECSS-E-ST-40_0860130

a. The customer shall verify that the requirements baseline, including the interface requirements document:

1. specifies a clear description of the environment in which the software operates;
2. specifies the characteristics of all external systems in interaction with the software product;
3. specifies the controllability and observability points for each application;
4. specifies the fault detection, identification, and recovery strategy to be implemented, and that the strategy is coherent with the dependability and safety level of the software under consideration;
5. specifies the modes/submodes and transition between modes: modes automaton;
6. specifies telemetries date management occurrences;
7. identifies the configuration data of the software;
8. identifies and justifies the margins policy in terms of memory and processing unit allocation;
9. defines operational scenario;
10. includes consistent and verifiable requirements;
11. is coherent with the results of the security analysis.

EXPECTED OUTPUT: *Requirements baseline verification report [DJF, SVR; SRR].*

NOTE Example of external systems are buses, computers, ground interfaces.

5.8.3.2 Verification of the technical specification

ECSS-E-ST-40_0860131

a. The supplier shall verify the technical specification including the interface control document ensuring that:

1. software requirements and interface are externally and internally consistent (not implying formal proof consistency);
2. the traceability between system requirements and software requirements is complete;
3. the software requirements that are not traced to the system requirements allocated to software are justified;
4. software requirements are verifiable;
5. software design is feasible;
6. operations and maintenance are feasible;
7. the software requirements related to safety, security, and criticality are correct;
8. the hardware environment constraints are identified;
9. the implementation constraints are identified;
10. the requirement verification method as specified in ECSS-Q-ST-80 clause 7.2.1.3 is feasible;
11. the logical model has been checked.

EXPECTED OUTPUT: The following outputs are expected:

a. Requirements traceability matrices [DJF, SVR or (SRS and ICD); PDR];

b. Requirements verification report [DJF, SVR; PDR].

5.8.3.3 Verification of the software architectural design

ECSS-E-ST-40_0860132

a. The supplier shall verify the architecture of the software item and the interface design ensuring that:

1. architecture and interface are externally consistent with the requirements of the software item;
2. there is internal consistency between the software components;
3. the traceability between the requirements and the software components is complete;
4. the software components that are not traced to the software requirements are justified;
5. producing a detailed design is feasible;
6. operations and maintenance are feasible;
7. the design is correct with respect to the requirements and the interfaces, including safety, security and other critical requirements;

8. the design implements proper sequence of events, inputs, outputs, interfaces, logic flow, allocation of timing and sizing budgets, and error handling;

9. the hierarchical breakdown from high level components to terminal ones is provided;

10. the dynamic features including tasks definition and priorities, synchronization mechanisms, shared resources management are provided and the real-time choices are justified;

11. the synchronisation between external interface and internal timing is achieved.

EXPECTED OUTPUT: The following outputs are expected:

a. Software architectural design to requirements traceability matrices [DJF, SVR or SDD; PDR];

b. Software architectural design and interface verification report [DJF, SVR; PDR].

5.8.3.4 Verification of the software detailed design

ECSS-E-ST-40_0860133

a. The supplier shall verify the software detailed design ensuring that:

1. detailed design is externally consistent with the architecture;

2. there is internal consistency between software components and software units;

3. the traceability between the architecture and the detailed design is complete;

4. the software units that are not traced to the components are justified;

5. testing is feasible, by assessing that:

   (a) controllability and observability features are identified and included in the detailed design in order to prepare the effective testing of the performance requirements;

   (b) computational invariant properties and temporal properties are added within the design;

   (c) fault injection is possible.

6. operation and maintenance are feasible;

7. the design is correct with respect to requirements and interfaces, including safety, security, and other critical requirements;

8. the design implements proper sequence of events, inputs, outputs, interfaces, logic flow, allocation of timing and sizing budgets, and error handling;

9. the design model has been checked;

10. the hierarchical breakdown from high level components to terminal ones is provided;

11. the dynamic features including tasks definition and priorities, synchronization mechanisms, shared resources management are provided and the real-time choices are justified;

12. the synchronisation between external interface and internal timing is achieved.

EXPECTED OUTPUT: *The following outputs are expected:*

a. Detailed design traceability matrices [DJF, SVR or SDD; CDR];

b. Detailed design verification report [DJF, SVR; CDR].

5.8.3.5 Verification of code

ECSS-E-ST-40_0860134

a. The supplier shall verify the software code ensuring at least that:

1. the code is externally consistent with the requirements and design of the software item;

2. there is internal consistency between software units;

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;

5. the code implements proper events sequences, consistent interfaces, correct data and control flow, completeness, appropriate allocation of timing and sizing budgets;

6. the code implements safety, security, and other critical requirements correctly as shown by appropriate methods;

7. the code is implemented in a way that it cannot result in runtime errors;

8. the effects of any residual runtime errors are controlled through error handling.

EXPECTED OUTPUT: *The following outputs are expected:*

a. Software code traceability matrices [DJF, SVR; CDR];

b. Software code verification report [DJF, SVR; CDR].

NOTE Examples of code checks are proposed in Annex T.

ECSS-E-ST-40_0860135

b. The supplier shall verify that the following code coverage is achieved

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

EXPECTED OUTPUT: *Code coverage verification report [DJF, SVR; CDR, QR, AR].*

NOTE This requirement is met by running unit, integration and validation tests, measuring the code coverage, and achieving the code coverage by additional (requirement based) tests, inspection or analysis.

ECSS-E-ST-40_0860137

c. The supplier shall measure code coverage by analysis of the results of the execution of tests.

EXPECTED OUTPUT: *Code coverage verification report [DJF, SVR; CDR, QR, AR].*

ECSS-E-ST-40_0860138

d. If it can be justified that the required percentage cannot be achieved by test execution, then analysis, inspection or review of design shall be applied to the non-covered code.

AIM: The goal of the complementary analysis is to assess that the non-covered code behaviour is as expected.

EXPECTED OUTPUT: *Code coverage verification report [DJF, SVR; CDR, QR, AR].*

ECSS-E-ST-40_0860139

e. In case the traceability between source code and object code cannot be verified, the supplier shall perform additional code coverage analysis on object code level as follows:

Code coverage VS. criticality category	**A**	**B**	**C**	**D**
Object code coverage	100%	N/A	N/A	N/A
NOTE: N/A means not applicable.				

EXPECTED OUTPUT: *Code coverage verification report [DJF, SVR; CDR, QR, AR].*

NOTE The use of some compiler optimization options can make the traceability between source code and object code not possible.

ECSS-E-ST-40_0860141

f. The supplier shall verify source code robustness.

AIM: use static analysis for the errors that are difficult to detect at run-time.

EXPECTED OUTPUT: *Robustness verification report [DJF, SVR; CDR].*

5.8.3.6 Verification of software unit testing (plan and results)

ECSS-E-ST-40_0860142

a. The supplier shall verify the unit tests results ensuring that:

1. the unit tests are consistent with detailed design and requirements;
2. the unit tests are traceable to software requirements, design and code;
3. software integration and testing are feasible;
4. operation and maintenance are feasible;
5. all activities defined in clause 5.5.3 are performed;
6. test results conform to expected results;
7. test results, test logs, test data, test cases and procedures, and test documentation are maintained under configuration management;
8. normal termination, i.e. the test end criteria defined in the unit test plan, is achieved;
9. abnormal termination of testing process is reported;
10. abnormal termination condition is documented in summary section of the unit test report, together with the unfinished testing and any uncorrected faults.

EXPECTED OUTPUT: *The following outputs are expected:*

a. Software unit tests traceability matrices [DJF, SVR; CDR];

b. Software unit testing verification report [DJF, SVR; CDR].

NOTE The trace to requirements is used to design the unit test cases in order to predict meaningful expected results.

5.8.3.7 Verification of software integration

ECSS-E-ST-40_0860143

a. The supplier shall verify that the integration has been performed according to the strategy specified in the software integration test plan, and the integration activities ensuring:

1. traceability to software architectural design;
2. internal consistency;

3. interface testing goals;

4. conformance to expected results.

EXPECTED OUTPUT: *Software integration verification report [DJF, SVR; CDR].*

5.8.3.8 Verification of software validation with respect to the technical specifications and the requirements baseline

ECSS-E-ST-40_0860144

a. The supplier shall verify the software validation results ensuring that the test requirements, test cases, test specifications, analysis, inspection and review of design cover all software requirements of the technical specification and the requirements baseline.

EXPECTED OUTPUT: *The following outputs are expected:*

a. Traceability of the requirements baseline to the validation specification [DJF, SVR or SVS; QR, AR];

b. Traceability of the technical specification to the validation specification [DJF, SVR or SVS; CDR].

ECSS-E-ST-40_0860145

b. The supplier shall verify the software validation results ensuring conformance to expected results.

EXPECTED OUTPUT: *The following outputs are expected:*

a. Validation report evaluation with respect to the technical specification [DJF, SVR; CDR];

b. Validation report evaluation with respect to the requirements baseline [DJF, SVR; QR].

5.8.3.9 Evaluation of validation: complementary system level validation

ECSS-E-ST-40_0860146

a. The supplier shall identify the requirements of the technical specification and the requirements baseline that cannot be tested in its own environment, and shall forward to the customer a request to validate them at system level.

EXPECTED OUTPUT: *Complement of validation at system level [DJF, SValP; PDR].*

NOTE For example: Some of the requirements cannot be verified because the test environment used for the validation does not allow it. These requirements can only be tested when the software is integrated within the system (e.g. satellite and launcher).

5.8.3.10 Verification of software documentation

ECSS-E-ST-40_0860147

a. The supplier shall verify the software documentation ensuring that:

1. the documentation is adequate, complete, and consistent;
2. documentation preparation is timely;
3. configuration management of documents follows specified procedures.

EXPECTED OUTPUT: Software documentation verification report [DJF, SVR; PDR, CDR, QR].

5.8.3.11 Schedulability analysis for real-time software

ECSS-E-ST-40_0860148

a. As part of the verification of the software requirements and architectural design, the supplier shall use an analytical model to perform a schedulability analysis and prove that the design is feasible.

EXPECTED OUTPUT: Schedulability analysis [DJF, SVR; PDR].

NOTE 1 Modelling and simulation can be used if it can be demonstrated that no analytical model exists.

NOTE 2 The schedulability analysis proves that the real–time behaviour is predictable, i.e. that all the tasks complete before their deadline in the worst case condition.

ECSS-E-ST-40_0860149

b. As part of the verification of the software detailed design, the supplier shall refine the schedulability analysis performed during the software architectural design on the basis of the software detailed design documentation.

EXPECTED OUTPUT: Schedulability analysis (update) [DJF, SVR; CDR].

ECSS-E-ST-40_0860150

c. As part of the verification of the software coding and testing, the supplier shall update the schedulability analysis performed during the software detailed design with the actual information extracted from the code.

EXPECTED OUTPUT: Schedulability analysis (update) [DJF, SVR; CDR].

ECSS-E-ST-40_0860739

d. As part of the verification of the software coding, testing and validation, the supplier shall update the schedulability analysis with the values measured on a representative operational and non-intrusive environment.

EXPECTED OUTPUT: Schedulability analysis (update) [DJF, SVR; QR].

5.8.3.12 Technical budgets management

ECSS-E-ST-40_0860151

a. As part of the verification of the software requirements and architectural design, the supplier shall estimate the technical budgets including memory size, PU utilization and the way the deadline are met.

EXPECTED OUTPUT: *Technical budgets - memory and PU estimation [DJF, SVR; PDR].*

ECSS-E-ST-40_0860152

b. As part of the verification of the software detailed design, the supplier shall update the estimation of the technical budgets.

EXPECTED OUTPUT: *Technical budgets (update) - memory and PU estimation [DJF, SVR; CDR].*

ECSS-E-ST-40_0860153

c. As part of the verification of the coding, testing and validation, the technical budgets shall be updated with the measured values and shall be compared to the margins.

EXPECTED OUTPUT: *Technical budgets (update) - memory and PU calculation [DJF, SVR; CDR, QR, AR].*

5.8.3.13 Behaviour modelling verification

ECSS-E-ST-40_0860154

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].*

ECSS-E-ST-40_0860155

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].*

ECSS-E-ST-40_0860156

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].*

5.9 Software operation process

5.9.1 Overview

This process consists of the following activities:

- process implementation;
- operational testing;
- software operation support.
- user support

5.9.2 Process implementation

5.9.2.1 Operational testing definition

ECSS-E-ST-40_0860157

a. The SOS entity shall establish procedures for:

1. testing the software product in its operational environment;
2. entering problem reports and modification requests to the maintenance process (see clause 5.10);
3. releasing the software product for operational use in accordance with the change control established and maintained in conformance with ECSS-M-ST-40;
4. vulnerability analysis and penetration testing.

EXPECTED OUTPUT: *Software operation support plan - operational testing specifications [OP, - ; ORR].*

NOTE Bullet 4 is relevant for software where there is a perceived risk of a security threat.

5.9.2.2 Software operation support plans and procedures development

ECSS-E-ST-40_0860158

a. The SOS entity shall complement the software user manual with the additional plans and procedures necessary to support the operation of the software and to perform the user support.

EXPECTED OUTPUT: *Software operation support plan - plans and procedures [OP, - ; ORR].*

5.9.2.3 Problem handling procedures definition

ECSS-E-ST-40_0860159

a. The SOS entity shall establish procedures for receiving, recording, resolving, tracking problems, and providing feedback.

EXPECTED OUTPUT: *Software operation support plan - procedures for problem handling [OP, - ; ORR].*

NOTE ECSS-Q-ST-80 clause 5.2.6 (nonconformances) and clause 5.2.5 (software problems) contain further requirements relevant for this clause.

ECSS-E-ST-40_0860740

b. The SOS entity shall ensure that information regarding problems that can have an impact on security is protected.

5.9.3 Operational testing

5.9.3.1 Operational testing execution

ECSS-E-ST-40_0860160

a. For each release of the software product, the SOS entity shall perform operational testing in accordance with the applicable procedures.

EXPECTED OUTPUT: Operational testing results [OP, - ; ORR].

5.9.3.2 Software operational requirements demonstration

ECSS-E-ST-40_0860161

a. The customer shall ensure that, prior to the first operations, the software is capable of implementing the operational requirements, testing the software in the following conditions:

1. the operating hardware environment,
2. the cases in which the software is designed to be fault tolerant,
3. the system configuration,
4. the sequence of operations;
5. the SOS entity interventions.

EXPECTED OUTPUT: Operational testing results [OP, - ; ORR].

NOTE This demonstration can be part of the acceptance tests of the system.

5.9.3.3 Software release

ECSS-E-ST-40_0860162

a. The software product shall be released for operational use in accordance with the procedure defined as per clause 5.9.2.1.

EXPECTED OUTPUT: Software product [DDF, - ; ORR].

5.9.4 Software operation support

5.9.4.1 Software operation support performance

ECSS-E-ST-40_0860163

a. The SOS entity shall execute the software operation support plan.

5.9.4.2 Problem handling

ECSS-E-ST-40_0860164

a. Encountered problems shall be recorded and handled in accordance with the applicable procedures.

EXPECTED OUTPUT: *Problem and non-conformance report [OP, - ; -].*

5.9.4.3 Vulnerabilities in operations

ECSS-E-ST-40_0860741

a. During operations, security vulnerabilities, threats and exploits shall be:

1. continuously monitored;
2. subject to further security analysis when evaluated relevant to the security of the system;
3. maintained for auditing purposes even when evaluated not relevant to the security of the system.

EXPECTED OUTPUT: *Software security analysis report and list of vulnerabilities [SF, -; -].*

NOTE The activities are performed in accordance with the applicable procedures defined in clause 5.9.2.1.

5.9.5 User support

5.9.5.1 Assistance to the user

ECSS-E-ST-40_0860165

a. The SOS entity shall provide assistance and consultation to the users.

EXPECTED OUTPUT: *User's request record - user's request and subsequent actions [OP, - ; -].*

ECSS-E-ST-40_0860166

b. The SOS entity shall record and monitor user's requests and subsequent actions.

EXPECTED OUTPUT: *User's request record - user's request and subsequent actions [OP, - ; -].*

5.9.5.2 Handling of user’s requests

ECSS-E-ST-40_0860167

a. The SOS entity shall forward user requests to the maintenance process for resolution.

EXPECTED OUTPUT: User’s request record - actions [OP, - ; -].

ECSS-E-ST-40_0860168

b. The SOS entity shall address user’s requests.

EXPECTED OUTPUT: User’s request record - actions [OP, - ; -].

ECSS-E-ST-40_0860169

c. The SOS entity shall report to the originators of the requests the actions that are planned and taken.

EXPECTED OUTPUT: User’s request record - actions [OP, - ; -].

5.9.5.3 Provisions of work–around solutions

ECSS-E-ST-40_0860170

a. If a reported problem has a temporary work–around solution before a permanent solution can be released, the SOS entity shall give to the originator of the problem report the option to use it.

EXPECTED OUTPUT: User’s request record - work around solution [OP, - ; -].

ECSS-E-ST-40_0860171

b. Permanent corrections, releases that include previously omitted functions or features, and system improvements shall be applied to the operational software product using the maintenance process as specified in clause 5.10.

EXPECTED OUTPUT: User’s request record - work around solution [OP, - ; -].

5.10 Software maintenance process

5.10.1 Overview

This process consists of the following activities:

- process implementation;
- problem and modification analysis;
- modification implementation;
- conducting maintenance reviews;
- software migration;
- software retirement.

5.10.2 Process implementation

5.10.2.1 Establishment of the software maintenance process

ECSS-E-ST-40_0860172

a. The maintainer shall develop, document, and execute plans and procedures for conducting the activities and tasks of the maintenance process.

EXPECTED OUTPUT: Maintenance plan - plans and procedures [MF, SMP ; QR, AR, ORR].

ECSS-E-ST-40_0860173

b. Software maintenance shall be performed using the same or fully compatible procedures, methods, tools and standards as used for the development.

EXPECTED OUTPUT: Maintenance plan - applicability of development process procedures, methods, tools and standards [MF, SMP ; QR, AR, ORR].

ECSS-E-ST-40_0860174

c. The maintainer shall implement (or establish the organizational interface with) the configuration management process (ECSS-M-ST-40) for managing modifications.

EXPECTED OUTPUT: Maintenance plan - configuration management process [MF, SMP ; QR, AR, ORR].

ECSS-E-ST-40_0860175

d. The maintainer shall establish procedures for receiving, recording and tracking problem reports and modification requests, providing feedback to the requester.

EXPECTED OUTPUT: Maintenance plan - problem reporting and handling [MF, SMP ; QR, AR, ORR].

ECSS-E-ST-40_0860176

e. Whenever problems are encountered, they shall be recorded and entered in accordance with the change control established and maintained in conformance with ECSS-M-ST-40.

EXPECTED OUTPUT: Problem and nonconformance report [MF, - ; QR].

NOTE ECSS-Q-ST-80 clause 5.2.6 (non-conformances) and clause 5.2.5 (software problems) contain further requirements relevant for this clause.

ECSS-E-ST-40_0860742

f. The software maintenance process shall address:

1. the constraints derived from the security sensitivity of the system,
2. the applicable measures from the risk treatment plan.

5.10.2.2 Long term maintenance for flight software

ECSS-E-ST-40_0860177

a. The maintainer shall propose solutions to be able to implement and upload modifications to the spacecraft up to its end of life.

EXPECTED OUTPUT: *Maintenance plan - long term maintenance solutions [MF, SMP ; QR, AR, ORR].*

5.10.3 Problem and modification analysis

5.10.3.1 Analysis of problem report and modification requests

ECSS-E-ST-40_0860178

a. The maintainer shall analyse the problem report or modification requests for its impact on the organization, the existing system, and the interfacing systems for the following:

1. type,
2. scope,
3. criticality.

EXPECTED OUTPUT: *Modification analysis report and problem analysis report [MF, - ; -].*

NOTE 1 Examples of type are corrective, improvement, preventive, or adaptive to new environment.

NOTE 2 Examples of scope are size of modification, cost involved, and time to modify.

NOTE 3 Examples of criticality are impact on performance, safety, or security.

ECSS-E-ST-40_0860179

b. The maintainer shall reproduce or verify the problem.

EXPECTED OUTPUT: *Modification analysis report and problem analysis report [MF, - ; -].*

ECSS-E-ST-40_0860180

c. Based upon the analysis, the maintainer shall develop options for implementing the modification.

EXPECTED OUTPUT: *Modification analysis report and problem analysis report [MF, - ; -].*

ECSS-E-ST-40_0860181

d. The maintainer shall document the problem or the modification request, the analysis results, the priorities in terms of operation needs, risk and effort and implementation options in the problem analysis report or in the modification analysis report, respectively.

EXPECTED OUTPUT: *Modification analysis report and problem analysis report [MF, - ; -].*

ECSS-E-ST-40_0860182

e. The maintainer shall obtain approval for the selected modification option in accordance with procedures agreed with the customer.

EXPECTED OUTPUT: Modification approval [MF; -].

5.10.4 Modification implementation

5.10.4.1 Analysis and documentation of product modification

ECSS-E-ST-40_0860183

a. The maintainer shall conduct and document an analysis to determine which documentation, models, software units, and their versions shall be modified.

EXPECTED OUTPUT: Modification documentation [MF, - ; -].

5.10.4.2 Documentation of software product changes

ECSS-E-ST-40_0860184

a. All changes to the software product shall be documented in accordance with the procedures for document control and configuration management.

EXPECTED OUTPUT: Modification documentation [MF, - ; -].

5.10.4.3 Invoking of software engineering processes for modification implementation

ECSS-E-ST-40_0860185

a. The maintainer shall apply the software engineering processes specified in clauses 5.3 to 5.8 and 5.11 that are relevant to the scope of the modifications, using the tailoring applied during the development of the software.

EXPECTED OUTPUT: Modification documentation [MF, - ; -].

ECSS-E-ST-40_0860186

b. Test and evaluation criteria for testing and evaluating the modified and the unmodified parts (models, software units, components, and configuration items) of the system shall be defined and documented.

EXPECTED OUTPUT: Modification documentation [MF, - ; -].

ECSS-E-ST-40_0860187

c. The complete and correct implementation of the new and modified requirements shall be ensured.

EXPECTED OUTPUT: Modification documentation [MF, - ; -].

ECSS-E-ST-40_0860188

d. It also shall be ensured that the original, unmodified requirements have not been affected.

EXPECTED OUTPUT: Modification documentation [MF, - ; -].

ECSS-E-ST-40_0860189

e. The test results shall be documented.

EXPECTED OUTPUT: Modification documentation [MF, - ; -].

5.10.5 Conducting maintenance reviews

5.10.5.1 Maintenance reviews

ECSS-E-ST-40_0860190

a. The maintainer shall conduct joint reviews with the organization authorizing the modification to determine the integrity of the modified system.

EXPECTED OUTPUT: Joint review reports [MF, - ; -].

5.10.5.2 Baseline for change

ECSS-E-ST-40_0860191

a. Upon successful completion of the reviews, a baseline for the change shall be established.

EXPECTED OUTPUT: Baseline for changes [MF, - ; -].

5.10.6 Software migration

5.10.6.1 Applicability of this Standard to software migration

ECSS-E-ST-40_0860192

a. If a system or software product is migrated from an old to a new operational environment, clauses 5.3 to 5.8 and 5.11 shall apply to the processes instantiated in the migration activities.

EXPECTED OUTPUT: Migration plan [MF, - ; -].

5.10.6.2 Migration planning and execution

ECSS-E-ST-40_0860193

a. A migration plan shall be developed, documented, and executed, including the following items:

1. requirements analysis and definition of migration;
2. development of migration tools;
3. conversion of software product and data;

4. migration execution;
5. migration verification;
6. support for the old environment in the future;
7. operator involvement in the activities.

EXPECTED OUTPUT: Migration plan [MF, - ; -].

5.10.6.3 Contribution to the migration plan

ECSS-E-ST-40_0860194

a. The maintainer shall contribute to the migration plan and justification including the following items:

1. statement of why the old environment is no longer to be supported;
2. description of the new environment with its date of availability;
3. description of other support options available, once support for the old environment has been removed;
4. the date as of which the transition takes place.

EXPECTED OUTPUT: Migration plan [MF, - ; -].

5.10.6.4 Preparation for migration

ECSS-E-ST-40_0860195

a. If parallel operations of the old and new environments are conducted for transition to the new environment, training shall be provided and specified in the operational plan.

EXPECTED OUTPUT: Migration plan [MF, - ; -].

5.10.6.5 Notification of transition to migrated system

ECSS-E-ST-40_0860196

a. When the scheduled migration takes place, notification shall be sent to all parties involved.

EXPECTED OUTPUT: Migration notification [MF, - ; -].

ECSS-E-ST-40_0860197

b. All associated old environment's documentation, logs, and code shall be placed in archives.

EXPECTED OUTPUT: Migration notification [MF, - ; -].

5.10.6.6 Post–operation review

ECSS-E-ST-40_0860198

a. A post–operation review shall be performed to assess the impact of changing to the new environment.

EXPECTED OUTPUT: Post operation review report [OP, - ; -].

ECSS-E-ST-40_0860199

b. The results of the review shall be sent to the appropriate authorities for information, guidance, and action.

EXPECTED OUTPUT: *Post operation review report [OP, - ; -].*

5.10.6.7 Maintenance and accessibility of data of former system

ECSS-E-ST-40_0860200

a. Data used by or associated with the old environment shall be accessible in accordance with the requirements for data protection and audit applicable to the data.

EXPECTED OUTPUT: *Migration plan [MF, - ; -].*

5.10.7 Software retirement

5.10.7.1 Retirement planning

ECSS-E-ST-40_0860201

a. Upon customer's request to retire a software product, a retirement plan to remove active support by the operator and maintainer shall be developed, documented and executed, ensuring:

1. cessation of full or partial support after the period of time specified by the customer;
2. archiving of the software product and its associated documentation;
3. responsibility for any future residual support issues;
4. transition to the new software product;
5. accessibility of archive copies of data, ensuring that adequate security of the data is maintained.

EXPECTED OUTPUT: *Retirement plan [MF, - ; -].*

5.10.7.2 Notification of retirement plan

ECSS-E-ST-40_0860202

a. The maintainer shall notify the retirement plan and related activities, including the following items:

1. description of the replacement or upgrade with its date of availability;
2. statement of why the software product is no longer to be supported;
3. description of other support options available, once support is removed.

EXPECTED OUTPUT: *Retirement notification [MF, - ; -].*

5.10.7.3 Identification of requirements for software retirement

ECSS-E-ST-40_0860203

a. If parallel operations of the retiring and the new software product are conducted for transition to the new system, user training shall be provided as specified in the business agreement.

EXPECTED OUTPUT: Retirement plan [MF, - ; -].

5.10.7.4 Maintenance and accessibility to data of the retired product

ECSS-E-ST-40_0860204

a. Data used by or associated with the retired software product shall be accessible in accordance with the business agreement requirements for data protection and audit applicable to the data.

EXPECTED OUTPUT: Retirement plan [MF, - ; -].

5.11 Software security process

5.11.1 Overview

The process consists of:

- Process implementation
- Software security analysis
- Security risk treatment
- Security activities in the software lifecycle

5.11.2 Process implementation

ECSS-E-ST-40_0860743

a. A software security management plan shall be produced documenting:

1. the security objectives for the project;
2. the organisation, roles and responsibilities for activities to be carried out in relation to security;
3. the relation between supplier organisation internal security processes and those of the project;
4. the stakeholders involved in security aspects of the project and their interfaces;
5. the planning and scheduling of security activities aligned with the project plan;

6. the activities to carry out and their associated outputs in terms of deliverables in relation to security;

7. the tools, methods and procedures to be used, including those for identification of new potential threats, vulnerabilities and potential sources of information and identification mechanisms;

8. establish the frequency for reviewing and updating the software security analysis and the risk treatment measures.

EXPECTED OUTPUT: Software security management plan [SF,SSMP; SRR].

NOTE 1 In some cases specific national, international or organisational regulations are applicable.

NOTE 2 Depending on the complexity of the development or project, a dedicated board (cyberboard) can be established to manage the software security process.

NOTE 3 The software security management plan can either be part of the software product assurance plan, if agreed with the customer, or a separate, dedicated plan.

ECSS-E-ST-40_0860744

b. The software security management plan shall be consistent with any higher level project or organisation security management plan.

ECSS-E-ST-40_0860745

c. A security manager with appropriate experience and training shall be appointed.

NOTE The role of the security manager can be shared with another role in the project if agreed with the customer.

ECSS-E-ST-40_0860746

d. The security constraints on the software lifecycle definition and implementation shall be assessed as part of the project planning.

NOTE Security constraints can be, for example, the need to restrict access to data or software artefacts to authorised individuals, or the application of organisational, national or international regulations.

5.11.3 Software security analysis

ECSS-E-ST-40_0860747

a. A software security analysis and corresponding risk treatment plan shall be produced at the start of the software product lifecycle and maintained up to the end of the lifecycle.

EXPECTED OUTPUT: Software security analysis report [SF, SSAR; SRR].

ECSS-E-ST-40_0860748

b. The methods to be used for the security analysis shall be identified as part of the planning of the project.

EXPECTED OUTPUT: Software security management plan [SF, SSMP; SRR].

ECSS-E-ST-40_0860749

c. The security analysis shall be produced, initially, on the basis of higher level requirements, security objectives and applicable regulations.

ECSS-E-ST-40_0860750

d. Where the software is part of a system, the security analysis shall be based on the security analysis from the next higher product architectural level, if there is one.

EXPECTED OUTPUT: Software security analysis report [SF, SSAR; SRR].

NOTE Software that is not part of a system will already be at the top architectural level.

ECSS-E-ST-40_0860751

e. Where relevant, results of the software security analysis shall be fed back to higher-level security analyses to support their update and refinement.

ECSS-E-ST-40_0860752

f. The software security analysis shall be reviewed and updated, as relevant, after updates of the next-higher-level analysis.

EXPECTED OUTPUT: Software security analysis report [SF, SSAR; -].

ECSS-E-ST-40_0860753

g. The software security analysis results, and its updates, shall be provided to lower level suppliers, on a need to know basis, to support software security analyses at their respective levels.

ECSS-E-ST-40_0860754

h. The software security analysis shall be reviewed and updated as relevant, after updates of lower-level software security analyses.

EXPECTED OUTPUT: Software security analysis report [SF, SSAR; -].

ECSS-E-ST-40_0860755

i. Where a security risk is associated to an identified vulnerability, a mapping shall be established between them.

ECSS-E-ST-40_0860756

j. The consistency and validity of mapped security risks and vulnerabilities shall be checked after changes to either of them.

ECSS-E-ST-40_0860757

k. The software security analysis shall be reviewed and updated as necessary:

1. at lifecycle milestones,
2. on the basis of change requests and
3. on the basis of newly identified threats or vulnerabilities.

ECSS-E-ST-40_0860758

l. In the absence of an existing classification, the software security analysis shall classify software components according to their sensitivity.

EXPECTED OUTPUT: Software security analysis report [SF, SSAR; PDR, CDR].

NOTE A classification can be can be defined on the basis of a classification guide.

5.11.4 Security risk treatment

ECSS-E-ST-40_0860759

a. On the basis of the software security analysis, risk treatment measures shall be proposed, to address, at least, the identified security risks above a certain pre-defined risk level.

EXPECTED OUTPUT: Security risk treatment plan [SF, -; SRR, PDR, CDR, QR, AR, ORR].

NOTE 1 For example, acceptable risks might not warrant the definition of risk treatment measures.

NOTE 2 The pre-defined risk level is often called “risk appetite” in the security domain.

ECSS-E-ST-40_0860760

b. Security risks that can have an impact on the complete system or the project shall be transferred to the overall risk management process, respecting the applicable security constraints.

EXPECTED OUTPUT: Risk management information [MF, -; -].

ECSS-E-ST-40_0860761

c. Where risk management information is sensitive, appropriate measures shall be taken to ensure adequate protection of that information within the overall risk management process.

ECSS-E-ST-40_0860762

d. Software requirements that have been introduced as a result of risk treatment measures shall be traced to the corresponding measure.

ECSS-E-ST-40_0860763

e. The software security analysis, associated security risks and risk treatments shall be updated systematically ensuring their continued consistency with each other and their validity.

EXPECTED OUTPUT: Software security analysis report and risk treatment plan [SF, -; -].

ECSS-E-ST-40_0860764

f. Risk treatment measures shall be implemented, monitored and reviewed systematically and periodically.

5.11.5 Security activities in the software lifecycle

5.11.5.1 Security in the requirements baseline

ECSS-E-ST-40_0860765

a. The customer shall use, and revise if necessary, the security analysis to support the elicitation of system software requirements, taking into account the concept of operations of the system, to ensure the correctness and completeness of system security requirements allocated to the software.

EXPECTED OUTPUT: Security system requirements allocated to the software [RB, SSS; SRR].

ECSS-E-ST-40_0860766

b. The customer shall use the security analysis to specify security assurance requirements in accordance with ECSS-Q-ST-80.

EXPECTED OUTPUT: Security assurance requirements. [RB, SSS; SRR].

NOTE The security assurance requirements provide grounds for confidence that top level security objectives are adequately met, including that security functions perform correctly and that the strength of those functions is robust enough to counter both unintentional errors and intentional attacks representative of an identified level of attacker resource and capability.

ECSS-E-ST-40_0860767

c. The security assurance requirements shall determine the type and extent of security verification and validation activities, including testing, to be conducted.

NOTE Security verification and validation activities can include, for example fuzzing tests, computing resources made available for testing tools, methods utilised by penetration tests.

ECSS-E-ST-40_0860768

d. At SRR, the software security analysis and associated risk treatment measures shall be reviewed to ensure that security risks are properly identified and addressed.

EXPECTED OUTPUT: Software security analysis report and risk treatment plan [SF, -; SRR].

5.11.5.2 Security in the software requirements specification and architectural design

ECSS-E-ST-40_0860769

a. The software security analysis shall be used, and revised if necessary, in the production of the software requirements to ensure that the defined functionality is complete and addresses the identified security risks.

EXPECTED OUTPUT: Security software requirements [TS, SRS; PDR].

ECSS-E-ST-40_0860770

b. The software security analysis shall be extended and revised iteratively during the definition of the software architecture to ensure that identified security risks are addressed by the chosen architecture, and to identify security vulnerabilities in the architectural design.

EXPECTED OUTPUT: Software security analysis report [SF, SSAR; PDR].

NOTE Identified security vulnerabilities are an input to the architectural design process and may drive it in order to reduce or remove them.

ECSS-E-ST-40_0860771

c. At PDR, the software security analysis and associated risk treatment plan shall be reviewed to ensure that:

1. the impact of the chosen software architecture on the identified security risks is properly documented;
2. security risks have been re-evaluated in view of the defined functionality and chosen architecture; and
3. residual vulnerabilities in the chosen design have been properly identified and documented.

EXPECTED OUTPUT: Software security analysis report, risk treatment plan and list of vulnerabilities [SF, -; PDR].

NOTE List of vulnerabilities can be included in the risk treatment plan or presented in specific document.

ECSS-E-ST-40_0860772

d. The selected architecture shall be reviewed at PDR to ensure that it addresses the identified security risks and that the chosen design components do not introduce additional unacceptable security risks.

5.11.5.3 Security in the detailed design and implementation engineering

ECSS-E-ST-40_0860773

a. The software security analysis shall be used and revised during the definition of the software detailed design to ensure that identified security risks are addressed by the chosen design, and to identify possible additional security vulnerabilities.

EXPECTED OUTPUT: Software security analysis report [SF, -; CDR].

ECSS-E-ST-40_0860774

b. The software security analysis shall be used during verification and validation activities to evaluate iteratively residual vulnerabilities and to reassess security risks.

EXPECTED OUTPUT: Identified security vulnerabilities and risks [SF, -; CDR].

ECSS-E-ST-40_0860775

c. At CDR, the software security analysis and residual vulnerabilities shall be reviewed to ensure that:

1. the impact of the detailed design on the identified security risks is properly documented;
2. security risks have been re-evaluated in view of the chosen detailed design and results of validation against the TS;
3. residual vulnerabilities identified during the production of the software architecture have been re-assessed in view of the detailed design; and
4. possible new vulnerabilities introduced by the detailed design have been identified and documented.

EXPECTED OUTPUT: Software security analysis report, associated risk treatment plan and list of vulnerabilities [SF, -; CDR].

NOTE The aspects of the review related to the detailed design can be reviewed already at DDR, if one has been defined in the lifecycle.

ECSS-E-ST-40_0860776

d. The detailed design shall be reviewed at CDR or DDR to ensure that it addresses the identified security risks and that the chosen design components do not introduce additional unacceptable security risks.

5.11.5.4 Security in the validation process

ECSS-E-ST-40_0860777

a. Validation activities shall conform with applicable security organisational regulations and policies.

NOTE Regulations and policies can require, for example, that only authorised personnel perform certain activities or the use of protected facilities.

ECSS-E-ST-40_0860778

b. The TRR's shall include a review and approval of the security-related unit and integration test results.

EXPECTED OUTPUT: Approval of the security-related unit and integration test results [DJF, -; TRR].

NOTE The review focuses on the coverage of security-related requirements.

ECSS-E-ST-40_0860779

c. The TRR's shall include a confirmation that any security measures required for execution of tests are in place.

EXPECTED OUTPUT: Security authorisation to proceed with testing [MF, -; TRR].

ECSS-E-ST-40_0860780

d. At QR, the software security analysis shall be reviewed to ensure that:

1. security risks have been re-evaluated in view of the results from validation against the RB; and

2. security risk treatment measures have been properly implemented and completed according to the risk treatment plan.

EXPECTED OUTPUT: Software security analysis report [SF, -; QR].

5.11.5.5 Security in the delivery and acceptance process

ECSS-E-ST-40_0860781

a. The supplier shall ensure that the software security analysis is fully valid for the intended operational environment, as a pre-condition to delivery.

NOTE Differences between the development, validation and operational environments can lead to additional security vulnerabilities and risks. This can happen for any delivery, whether part of development or maintenance.

ECSS-E-ST-40_0860782

b. The software security analysis, residual security risks, vulnerabilities and associated risk treatment measures shall be reviewed at acceptance review.

ECSS-E-ST-40_0860783

c. At AR, the software security analysis and risk treatment measures status shall be reviewed to ensure:

1. that all risk treatment measures have been implemented and completed according to the risk treatment plan; and

2. the acceptability of the residual security risks.

EXPECTED OUTPUT: Joint review report [DJF, -; AR].

5.11.5.6 Security in the operations and maintenance process

ECSS-E-ST-40_0860784

a. The software security analysis shall be periodically reviewed during operations to ensure its continued validity with regard to changes in known security vulnerabilities, threats and exploits.

EXPECTED OUTPUT: Software security analysis report [SF, -; -].

NOTE New vulnerabilities, threats and exploits can be reported, in particular for third-party products, by user and developer communities or by external organisations.

ECSS-E-ST-40_0860785

b. The software security analysis shall be reviewed to ensure their continued validity in response to:

1. reported security-relevant events and incidents,

2. software problem reports, and

3. change requests.

EXPECTED OUTPUT: Software security analysis report [SF, -; -].

ECSS-E-ST-40_0860786

c. Trends in performance indicators, security events and incidents, and software problem reports shall be reviewed periodically to identify possible new security risks and re-evaluate existing ones.

ECSS-E-ST-40_0860787

d. The software security analysis, vulnerabilities and risks shall be reviewed and updated as necessary as part of changes implemented during software maintenance.

EXPECTED OUTPUT: Software security analysis report [SF, -; -].

ECSS-E-ST-40_0860788

e. The software security analysis shall be reviewed and updated as necessary, as part of migration, to ensure its validity in the new environment and to assess the impact on the planning and execution of the migration process.

EXPECTED OUTPUT: Software security analysis report [SF, -; -].

ECSS-E-ST-40_0860789

f. The software security analysis shall be reviewed and updated as necessary, prior to software retirement to ensure the security of the retirement, including secure data sanitation and secure data archiving.

EXPECTED OUTPUT: Software security analysis report [SF, -; -].

ECSS-E-ST-40_0860790

g. Data sanitation shall consider data in storage media and in hardware where data is persisted.

EXPECTED OUTPUT: List of items to be sanitised [SF, -; CDR].

NOTE Data can remain, for example, in permanent memory inside computers, which need to be disposed of appropriately.

ECSS-E-ST-40_0860791

h. Software, supporting systems and media shall be sanitised according to applicable organisational policies, regulations and standards.

NOTE This includes, for example, using approved equipment, techniques or procedures.

ECSS-E-ST-40_0860792

i. Records shall be kept of media and equipment sanitisation and disposal actions.

EXPECTED OUTPUT: Records of media and equipment sanitation [SF, -; -].

Annex A (informative) Software documentation

This annex defines the structure of the software documents to be produced as depicted in Figure A-1.

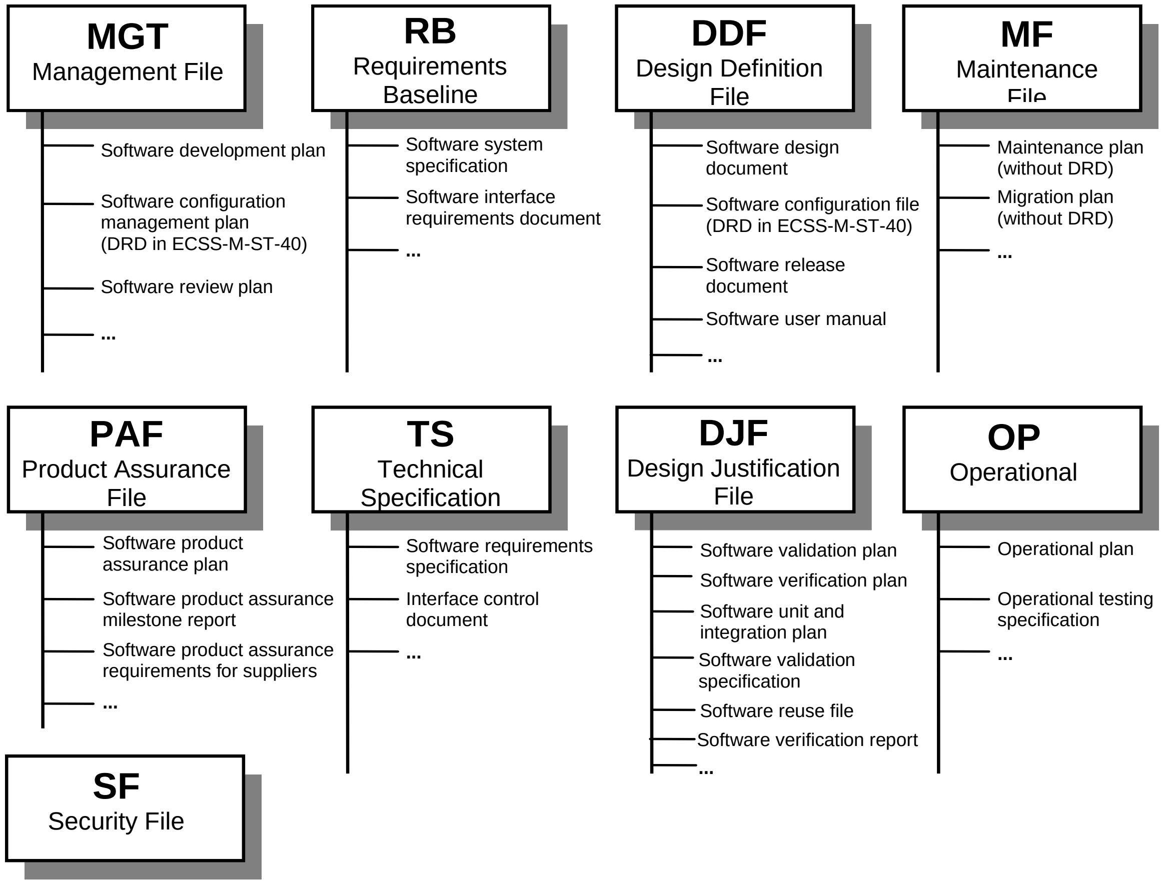
MGT
Management File
Software development plan
Software configuration management plan (DRD in ECSS-M-ST-40)
Software review plan
...
RB
Requirements Baseline
Software system specification
Software interface requirements document
...
DDF
Design Definition File
Software design document
Software configuration file (DRD in ECSS-M-ST-40)
Software release document
Software user manual
...
MF
Maintenance File
Maintenance plan (without DRD)
Migration plan (without DRD)
...
PAF
Product Assurance File
Software product assurance plan
Software product assurance milestone report
Software product assurance requirements for suppliers
...
TS
Technical Specification
Software requirements specification
Interface control document
...
DJF
Design Justification File
Software validation plan
Software verification plan
Software unit and integration plan
Software validation specification
Software reuse file
Software verification report
...
OP
Operational
Operational plan
Operational testing specification
...
SF
Security File

Figure A-1: Overview of software documents

Table A-1 represents the document requirements list identifying the software documentation to be produced in accordance with the requirements defined in this Standard and in ECSS-Q-ST-80 as output of the relevant processes.

Table A-1: ECSS-E-ST-40 and ECSS-Q-ST-80 Document requirements list (DRL)

Related file	DRL item (e.g. Plan, document, file, report, form, matrix)	DRL item having a DRD	SRR	PDR	CDR	QR	AR	ORR
RB	Software system specification (SSS)	ECSS-E-ST-40 Annex B	✓					
	Interface requirements document (IRD)	ECSS-E-ST-40 Annex C	✓					
	Safety and dependability analysis results for lower-level suppliers	-	✓					
TS	Software requirements specification (SRS)	ECSS-E-ST-40 Annex D		✓				
	Software interface control document (ICD)	ECSS-E-ST-40 Annex E		✓	✓			
DDF	Software design document (SDD)	ECSS-E-ST-40 Annex F		✓	✓			
	Software configuration file (SCF)	ECSS-M-ST-40 Annex E		✓	✓	✓	✓	✓
	Software release document (SRelD)	ECSS-E-ST-40 Annex G				✓	✓	
	Software user manual (SUM)	ECSS-E-ST-40 Annex H			✓	✓	✓	
	Software source code and media labels	-			✓			
	Software product and media labels	-				✓	✓	✓
	Training material	-				✓		
DJF	Software verification plan (SVerP)	ECSS-E-ST-40 Annex I	✓	✓				
	Software validation plan (SValP)	ECSS-E-ST-40 Annex J		✓				
	Independent software verification & validation plan	-	✓	✓				

Related file	DRL item (e.g. Plan, document, file, report, form, matrix)	DRL item having a DRD	SRR	PDR	CDR	QR	AR	ORR
	Software integration test plan (SUITP)	ECSS-E-ST-40 Annex K		✓	✓			
	Software unit test plan (SUITP)	ECSS-E-ST-40 Annex K			✓			
	Software validation specification (SVS) with respect to TS	ECSS-E-ST-40 Annex L			✓			
	Software validation specification (SVS) with respect to RB	ECSS-E-ST-40 Annex L				✓	✓	
	Acceptance test plan	-				✓	✓	
	Software unit test report	-			✓			
	Software integration test report	-			✓			
	Software validation report with respect to TS	-			✓			
	Software validation report with respect to RB	-				✓	✓	
	Acceptance test report	-					✓	
	Installation report	-					✓	
	Software verification report (SVR)	ECSS-E-ST-40 Annex M	✓	✓	✓	✓	✓	✓
	Independent software verification & validation report	-		✓	✓	✓	✓	✓
	Software reuse file (SRF)	ECSS-E-ST-40 Annex N	✓	✓	✓			
	Software problems reports and nonconformance reports	-	✓	✓	✓	✓	✓	✓
	Joint review reports	-	✓	✓	✓	✓	✓	

Related file	DRL item (e.g. Plan, document, file, report, form, matrix)	DRL item having a DRD	SRR	PDR	CDR	QR	AR	ORR
	Justification of selection of operational ground equipment and support services	-	✓	✓				
MGT	Software development plan (SDP)	ECSS-E-ST-40 Annex O	✓	✓				
	Software review plan (SRevP)	ECSS-E-ST-40 Annex P	✓	✓				
	Software configuration management plan	ECSS-M-ST-40 Annex A	✓	✓				
	Training plan	-	✓					
	Interface management procedures	-	✓					
	Identification of NRB SW and members	-	✓					
	Procurement data	-	✓	✓				
MF	Software maintenance plan	ECSS-E-ST-40 Annex T				✓	✓	✓
	Maintenance records	-				✓	✓	✓
	SPR and NCR - Modification analysis report - Problem analysis report - Modification documentation- Baseline for change - Joint review reports	-						
	Migration plan and notification	-						
	Retirement plan and notification	-						
OP	Software operation support plan	-						✓
	Operational testing results	-						✓
	SPR and NCR - User's request record - Post operation review report	-						✓

Related file	DRL item (e.g. Plan, document, file, report, form, matrix)	DRL item having a DRD	SRR	PDR	CDR	QR	AR	ORR
SF	Software security management plan (SSMP)	-	✓	✓				
	Software security analysis report (SSAR)	-	✓	✓	✓	✓	✓	
	Software analyses results for lower level suppliers							
	Security risk treatment plan (SRTP)	-	✓	✓	✓	✓	✓	✓
PAF	Software product assurance plan (SPAP)	ECSS-Q-ST-80 Annex B	✓	✓	✓	✓	✓	✓
	Software product assurance requirements for suppliers	-	✓					
	Audit plan and schedule	-	✓					
	Review and inspection plans or procedures	-						
	Procedures and standards	-		✓				
	Modelling and design standards	-	✓	✓				
	Coding standards and description of tools	-		✓				
	Software problem reporting procedure	-		✓				
	Software dependability and safety analysis report - Criticality classification of software components	-		✓	✓	✓	✓	
	Software product assurance report	-						
	Software product assurance milestone report (SPAMR)	ECSS-Q-ST-80 Annex C	✓	✓	✓	✓	✓	✓
	Statement of compliance with test plans and procedures	-			✓	✓	✓	✓
	Records of training and experience	-						

Related file	DRL item (e.g. Plan, document, file, report, form, matrix)	DRL item having a DRD	SRR	PDR	CDR	QR	AR	ORR
	(Preliminary) alert information	-						
	Results of pre-award audits and assessments, and of procurement sources	-						
	Software process assessment plan	-						
	Software process assessment records	-						
	Review and inspection reports	-						
	Receiving inspection report	-	✓	✓	✓	✓		
	Input to product assurance plan for systems operation	-						✓
NOTE:	Shaded boxes are the contributions of ECSS-Q-ST-80.							

Annex B (normative) Software system specification (SSS) - DRD

B.1 DRD identification

B.1.1 Requirement identification and source document

The software system specification (SSS) document is called from the normative provisions summarized in Table B-1.

Table B-1: SSS traceability to ECSS-E-ST-40 and ECSS-Q-ST-80 clauses

ECSS Standard	Clauses	DRD section
ECSS-E-ST-40	5.2.2.1a eo a	<5.2>
	5.2.2.1a eo b	<6.3> and <6.4>
	5.2.2.1a eo c	<5.11>
	5.2.2.1a eo d	<5.12>
	5.2.2.1a eo e	<5.12>
	5.2.2.1a eo f	<5.2>c.
	5.2.2.1a eo g	<5.6>
	5.2.2.1a eo h	<5.9>
	5.2.2.1b	<5.6>
	5.2.2.1c	<5.6>
	5.2.2.1d	<5.6>
	5.2.2.2	<5.13>
	5.2.2.3	<5.2>d.
	5.2.3.1a	<6.1>a
	5.2.3.1b	<6.1>b
	5.2.3.2	<6.2>
	5.2.3.3	<6.4>a.1.
	5.2.4.1	<6.4>a.2.
	5.2.4.2	<6.4>a.3.
	5.2.4.4	<5.4>
	5.2.4.5	<5.10>

	5.2.4.6	<5.2>e.
	5.2.4.7	<5.9>
	5.2.4.8	<5.7>, <5.8>
	5.2.4.9	<6.4>a.4.
	5.2.4.10	<5.14>
	5.2.4.11	<5.15>
	5.3.8.1	<5.5>
ECSS-Q-ST-80	7.1.1 eo a	<5.9>
	7.1.2 eo a	<5.9>
	7.2.1.1. eo a	<5.9>
	7.2.1.3 eo a	<5.1>c.
eo = Expected Output		

B.1.2 Purpose and objective

The software system specification contains the customer’s requirements. It is generated by the system engineering processes related to software. It is the highest level description of the software and, together with the interface requirements document, provides the criteria that are used to validate and accept the software.

The information about traceability to high–level requirements can be in the software system specification or in the requirements traceability in the design justification file. In either case a cross–reference is done.

The software system specification can be produced as a standalone document or as part of a system–level specification document. It can be included, for example, in the technical specification introduced by ECSS-E-ST-10-06. If produced as a standalone document, the present DRD applies, else the DRD described in ECSS-E-ST-10-06 for the establishment of a functional and technical specification applies.

The software system specification is a major component of the requirements baseline and is the primary input for the system requirements review (SRR)

The requirements baseline towards the supplier includes, at each recursive level, the level of requirements detail necessary for an unambiguous implementation by the supplier. As on subsequent contractual levels, the design becomes more mature, and the implementation freedom is consequently more limited. This is reflected in the specification of requirements towards subcontractors who therefore, at lower contractual levels, receive more (design, implementation) constraining requirements.

B.2 Expected response

B.2.1 Scope and content

<1> Introduction

ECSS-E-ST-40_0860205

a. The SSS shall contain a description of the purpose, objective, content and the reason prompting its preparation.

<2> Applicable and reference documents

ECSS-E-ST-40_0860206

a. The SSS shall list the applicable and reference documents to support the generation of the document.

<3> Terms, definitions and abbreviated terms

ECSS-E-ST-40_0860207

a. The SSS shall include any additional terms, definition or abbreviated terms used not already defined in any applicable or reference documentation.

<4> General description

<4.1> Product perspective

ECSS-E-ST-40_0860208

a. The SSS shall describe the product in perspective with other related systems.

ECSS-E-ST-40_0860209

b. If the product is to replace an existing system, the system shall be described and referenced.

<4.2> General capabilities

ECSS-E-ST-40_0860210

a. The SSS shall describe the main capabilities to be supported by the software.

NOTE Reference to state and mode of the system can be made.

<4.3> General constraints

ECSS-E-ST-40_0860211

a. The SSS shall describe any item that limits the supplier’s options for designing and developing the software.

<4.4> Operational environment

ECSS-E-ST-40_0860212

a. The SSS shall describe the software operational environment.

ECSS-E-ST-40_0860666

b. Context diagrams may support this narrative description to summarize external interfaces and system block diagrams to show how the activity fits within the larger system.

ECSS-E-ST-40_0860667

c. The nature of exchanges with external systems should be listed.

ECSS-E-ST-40_0860668

d. If a system specification defines a product that is a component of a parent system or project, then the SSS should list the activities that are supported by external systems.

ECSS-E-ST-40_0860216

e. References to the interface requirements documents that define the external interfaces with the other systems shall be provided

ECSS-E-ST-40_0860217

f. The computer infrastructure to be used shall be also described.

<4.5> Assumptions and dependencies

ECSS-E-ST-40_0860218

a. The SSS shall list or make reference to the existing assumptions that the specific requirements are based on.

NOTE 1 Risks analysis is used to identify assumptions that cannot prove to be valid.

NOTE 2 A constraint requirement, for example, can specify an interface with a system which does not exist.

NOTE 3 If the production of the system does not occur when expected, this system specification can change.

<5> Specific requirements

<5.1> General

NOTE The following provisions apply to the system specific requirements listed in the SSS, as specified in <5.2> to <5.12> below:

ECSS-E-ST-40_0860219

a. Each requirement shall be uniquely identified.

ECSS-E-ST-40_0860220

b. When requirements are expressed as models, the supplier shall establish a way to assign identifiers within the model for sake of traceability.

ECSS-E-ST-40_0860221

c. Each requirement shall be associated to a validation method and a version, as per 5.2.4.1a. and ECSS-Q-ST-80, 7.2.1.3.

<5.2> Capabilities requirements (5.2.2.1a.)

ECSS-E-ST-40_0860222

a. The SSS shall list the requirements specifying the system behaviour and its associated performances.

ECSS-E-ST-40_0860223

b. Description of capability requirements shall be organized by capability type.

NOTE For example, requirements can be organized per controlled subsystem (e.g. AOCS, power, and thermal).

ECSS-E-ST-40_0860224

c. Capability requirements shall include the real-time behaviour and constraints of the system (5.2.2.1f.)

ECSS-E-ST-40_0860225

d. The HMI capability requirements shall result from the human factor engineering process described in ECSS-E-ST-10-11 (5.2.2.3)

ECSS-E-ST-40_0860226

e. The SSS shall state the capabilities to be implemented by on-board control procedures (OBCP) (5.2.4.6)

<5.3> System interface requirements (5.2.4.3)

ECSS-E-ST-40_0860227

a. The SSS shall list, or refer to the IRD, any interface requirements imposed on the system.

ECSS-E-ST-40_0860228

b. The requirements of the following shall be either be listed in the SSS or referred to in the IRD:

1. communication interfaces,
2. hardware interfaces,
3. software interfaces,
4. HMI,

<5.4> Adaptation and missionization requirements (5.2.4.4)

ECSS-E-ST-40_0860229

a. The SSS shall list the data that can vary according to operational needs and any site–dependant data, including the system database.

<5.5> Computer resource requirements (5.3.8.1)

<5.5.1> Computer hardware resource requirements

ECSS-E-ST-40_0860230

a. The SSS shall list the requirements on the computer hardware to be used.

<5.5.2> Computer hardware resource utilization requirements

ECSS-E-ST-40_0860231

a. The SSS shall list the requirements on the computer resource utilization (e.g. processor capacity and memory capacity) available for the software item (e.g. sizing and timing).

<5.5.3> Computer software resource requirements

ECSS-E-ST-40_0860232

a. The SSS shall list requirements on the software items to be used by or incorporated into the system (or constituent software product) (e.g. a specific real time operating system)

<5.6> Security requirements (5.2.2.1g.)

ECSS-E-ST-40_0860233

a. The SSS shall list the security requirements applicable to the system.

<5.7> Safety requirements (5.2.4.8)

ECSS-E-ST-40_0860234

a. The SSS shall list the safety requirements applicable to the system.

<5.8> Reliability and availability requirements (5.2.4.8)

ECSS-E-ST-40_0860235

a. The SSS shall list the reliability and availability requirements applicable to the system.

<5.9> Quality requirements

ECSS-E-ST-40_0860236

a. The SSS shall list the quality requirements applicable to the software (e.g. usability, reusability (5.2.4.7), and portability), and the applicable software development standards (5.2.4.5)

NOTE When future reuse of developed software components is a requirement, the customer specifies the generic application domain of these components. This can, for example, include requirements on software architecture for given target computers and operating systems, the interfaces required for reuse and the level where reuse is required (e.g. function, sub–system, and code units).

<5.10> Design requirements and constraints (5.2.4.5)

ECSS-E-ST-40_0860237

a. The SSS shall include the requirements which constraint the design and production of the system.

NOTE When the software is integrated into a system, the customer can check for applicability of some harmonization constraints such as: specification of the operating system to be used, specification of COTS to be used (e.g. database and HMI generator), and specification of the SDE to be used.

ECSS-E-ST-40_0860238

b. At least, the following type of requirements shall be considered:

1. constraints on the software architecture,
2. utilization of standards,
3. utilization of existing components,
4. utilization of customer furnished components or COTS components,
5. utilization of design standard,
6. utilization of data standard,
7. utilization of a specific programming language,
8. utilization of specific naming convention,
9. flexibility and expansion, and
10. utilization of HMI standards.

NOTE 1 to item 4: For instance, a database can be used to produce automatically configured software (e.g. generation of tables, constant data, initial values).

NOTE 2 to item 4: For instance, a database can be, according to ECSS-E-ST-10 and ECSS-E-ST-70, e.g. a mission–operations database, an operational database of ground elements, a space segment database or a spacecraft database.

<5.11> Software operations requirements (5.2.2.1c.)

ECSS-E-ST-40_0860239

a. The SSS shall include the system requirements for the operation of the software.

NOTE The supplier’s response is provided in releasing the operational plan for execution as established in clause 5.9.

<5.12> Software maintenance requirements (5.2.2.1d., e.)

ECSS-E-ST-40_0860240

a. The SSS shall include the system requirements for the maintenance of the software (including the system requirements for in flight modifications capabilities).

NOTE 1 The supplier’s response is agreed with the customer in the system requirements review (SRR), intended to release the maintenance plan for execution as established in clause 5.10.

NOTE 2 Due to the long lifetimes often encountered with flight software, special requirements also exist to ensure that the supporting tools (e.g. compilers, engineering tools and inflight modification tools) can support the in–orbit reprogramming during the specified lifetime.

<5.13> System and software observability requirements (5.2.2.2)

ECSS-E-ST-40_0860241

a. The SSS shall include the system and software requirements for the observability of the system

NOTE 1 Observability requirements can impair the performance requirements (e.g. computer throughput, bandwidth, and ground exploitation rate of telecommands). This has an impact when specifying the observability requirements (e.g. considering the need of oversampling).

NOTE 2 In case of ACG the capability of a code generator to automatically instrument the code - and to remove such code parts - to cover observability requirements is considered.

<5.14> Security constraints for the software development and integration environment (5.2.4.10)

ECSS-E-ST-40_0860793

a. The SSS shall list the specific security constraints for the software development and integration environment.

<5.15> Secure software delivery requirements (5.2.4.11)

ECSS-E-ST-40_0860794

a. The SSS shall specify the requirements for secure software delivery.

<6> Verification, validation and system integration

<6.1> Verification and validation process requirements (5.2.3.1)

ECSS-E-ST-40_0860242

a. The SSS shall include the requirements needed for planning and setting up the system verification and validation process related to software

NOTE Specific customer requirements (e.g. availability or usage of customer furnished equipment like ETM, EM, FM and related ground systems) are needed in order to plan and to set up the system level verification and validation processes related to software. Furthermore, it is important to define for each testing or mission phase, or mission increment as far as applicable, any usage of hardware/software simulators, as representative flight and ground environments, for system payloads' and experiments' (pre-) integration testing, hardware/software compatibility as well for acceptance testing.

ECSS-E-ST-40_0860795

b. The SSS shall include the requirements needed for planning and setting up the system security verification and validation process related to software

<6.2> Validation approach (5.2.3.2)

ECSS-E-ST-40_0860243

a. The SSS shall include the requirements for the validation of the software against requirements baseline and technical specification, in particular mission representative data and scenarios, and operational procedures to be used.

<6.3> Validation requirements (5.2.2.1b.)

ECSS-E-ST-40_0860244

a. The SSS shall describe the validation requirements specified to support the demonstration that software requirements are met.

ECSS-E-ST-40_0860245

b. For each of the identified requirements in 5.2.2.1b., a validation method shall be included.

ECSS-E-ST-40_0860669

c. If a given requirement need not to be taken into account for the software validation against the requirements baseline, then it should be clearly identified.

NOTE A matrix (requirements to validation method correlation table) can be used to state the validation methods applicable to each requirement. This information can be further split into an information for validation requirements and an information for acceptance requirements, depending upon the project needs.

<6.4> Verification requirements (5.2.2.1b.)

ECSS-E-ST-40_0860247

a. The SSS shall include requirements for:

1. the installation and acceptance of the software (5.2.3.3),
2. the needed versions, their content and their medium (5.2.4.1),
3. the needed supplier support for system integration (5.2.4.2),
4. the format and delivery medium of all exchanged data in the project, including the interface and the system database (5.2.4.9).

<7> System models

ECSS-E-ST-40_0860670

a. The system models may be placed as appendix to the SSS if a specification language is used to support the system and software co-engineering processes in order to enable the generation of the software specification and architecture from the (formal) system specification

NOTE The system specification language intends to produce in particular the computational model, the data model, the functional model, the event model and the failure model. The models include at least the definition of the logical properties, the liveliness properties and the timeless properties. The models are exercised to verify the properties by schedulability analysis or model checking.

B.2.2 Special remarks

None.

Annex C (normative) Software interface requirements document (IRD) - DRD

C.1 DRD identification

C.1.1 Requirement identification and source document

The interface requirements document (IRD) document is called from the normative provisions summarized in Table C-1.

Table C-1: IRD traceability to ECSS-E-ST-40 and ECSS-Q-ST-80 clauses

ECSS Standard	Clauses	DRD section
ECSS-E-ST-40	5.2.4.3	All

C.1.2 Purpose and objective

The interface requirements document is a major component of the requirements baseline and is the primary input for the system requirements review (SRR).

The interface requirements document contains the customer’s requirements. It is generated by the system engineering processes related to software. It is the highest level description of the software and, together with the software system specification, provides the criteria that are used to validate and accept the software.

The information about traceability to high-level requirements is in the software system specification or in the requirements traceability in the design justification file. In either case a cross-reference is done.

The interface requirements document is produced as a standalone document or as part of a system-level specification document. It is included, for example, in the system technical specification introduced by ECSS-E-ST-10-06, for which the DRD described in ECSS-E-ST-10-06 for the establishment of a functional and technical specification applies, or it is produced as a standalone document, for which the present DRD applies.

C.2 Expected response

C.2.1 Scope and content

<1> Introduction

ECSS-E-ST-40_0860249

a. The IRD shall contain a description of the purpose, objective, content and the reason prompting its preparation.

<2> Applicable and reference documents

ECSS-E-ST-40_0860250

a. The IRD shall list the applicable and reference documents to support the generation of the document.

<3> Terms, definitions and abbreviated terms

ECSS-E-ST-40_0860251

a. The IRD shall include any additional terms, definition or abbreviated terms used not already defined in any applicable or reference documentation.

<4> General description

<4.1> Product perspective

ECSS-E-ST-40_0860252

a. The IRD shall describe the product external interfaces to other systems

<4.2> General constraints

ECSS-E-ST-40_0860253

a. The IRD shall describe any item that limits the supplier's options for designing and developing the interfaces of the software.

<4.3> Operational environment

ECSS-E-ST-40_0860671

a. Context diagrams may support this narrative description to summarize external interfaces and system block diagrams to show how the activity fits within the larger system.

ECSS-E-ST-40_0860672

b. The nature of exchanges with external systems should be listed.

ECSS-E-ST-40_0860673

c. If a system specification defines a product that is a component of a parent system or project, then the IRD should list the activities that are supported by external systems.

ECSS-E-ST-40_0860257

d. References to the interface control documents that define the external interfaces with the other systems shall be provided.

<4.4> Assumptions and dependencies

ECSS-E-ST-40_0860258

a. The IRD shall list or make reference to the existing assumptions and dependencies.

<5> Specific requirements

<5.1> General

ECSS-E-ST-40_0860259

a. Each requirement shall be uniquely identified.

<5.2> Capabilities requirements

ECSS-E-ST-40_0860260

a. The IRD shall list the external interface requirements specifying the interface behaviour and its associated performances.

<5.3> System interface requirements

ECSS-E-ST-40_0860261

a. The IRD shall define any interface requirements imposed on the system.

1. system level interface requirements,
2. software-hardware interfaces,
3. system level data interfaces,
4. communication interfaces,
5. hardware interfaces,
6. software interfaces,
7. HMI,
8. security aspects of the external interfaces.

<5.4> Adaptation / missionization requirements

ECSS-E-ST-40_0860262

a. The IRD shall list the external data that can vary according to operational needs and any site-dependant data.

<6> Validation requirements

ECSS-E-ST-40_0860263

a. The IRD shall describe if necessary the validation requirements specified to support the demonstration that software requirements are met.

ECSS-E-ST-40_0860264

b. For each of the identified requirements in <5>, a validation method shall be included.

ECSS-E-ST-40_0860674

c. If a given requirement need not be taken into account for the software validation against the requirements baseline, then it should be clearly identified.

NOTE A matrix (requirements to validation method correlation table) is used to state the validation methods applicable to each requirement. This information is further split into an information for validation requirements and an information for acceptance requirements, depending upon the project needs.

C.2.2 Special remarks

None.

Annex D (normative) Software requirements specification (SRS) - DRD

D.1 DRD identification

D.1.1 Requirement identification and source document

The software requirements specification (SRS) document is called from the normative provisions summarized in Table D-1.

Table D-1: SRS traceability to ECSS-E-ST-40 and ECSS-Q-ST-80 clauses

ECSS Standard	Clauses	DRD section
ECSS-E-ST-40	5.4.2.1 eo a	<4.2>, <5.2>, <5.3>, <5.6>
	5.4.2.1 eo b	<5.5>, <5.2>, <5.9>, <5.11>, <5.12>, <5.13>, <5.14>, <5.17>
	5.4.2.1 eo c	<5.10>
	5.4.2.1 eo d	<5.8>
	5.4.2.1 eo e	<5.16>
	5.4.2.1 eo f	<5.15>
	5.4.2.1 eo g	<6>
	5.4.2.1 eo i	<5.7>b.7.
	5.4.2.2	<5.7>b.5.
	5.4.2.3 eo a	<8>
	5.4.2.3 eo b	<8>
	5.4.2.3 eo c	<8>
	5.8.3.2 eo a	<7>, <5.1>c.
ECSS-Q-ST-80	6.3.2.4	<5>
	7.1.1 eo b	<5.10>
	7.1.2 eo b	<5.10>
	7.2.1.1 eo b	<5.10>
	7.2.1.3 eo b	<6>
eo = Expected Output		

D.1.2 Purpose and objective

The software requirements specification is a major constituent of the technical specification (TS). It describes the functional and non-functional requirements applicable to the software item.

D.2 Expected response

D.2.1 Scope and content

<1> Introduction

ECSS-E-ST-40_0860266

a. The SRS shall contain a description of the purpose, objective, content and the reason prompting its preparation.

<2> Applicable and reference documents

ECSS-E-ST-40_0860267

a. The SRS shall list the applicable and reference documents to support the generation of the document.

<3> Terms, definitions and abbreviated terms

ECSS-E-ST-40_0860268

a. The SRS shall include any additional terms, definition or abbreviated terms used not already defined in any applicable or reference documentation.

<4> Software overview

<4.1> Function and purpose

ECSS-E-ST-40_0860269

a. The SRS shall describe the purpose of the product.

<4.2> Environmental considerations

ECSS-E-ST-40_0860270

a. The SRS shall summarize:

1. the physical environment of the target system;
2. the hardware environment in the target system;
3. the operating environment in the target system;

<4.3> Relation to other systems

ECSS-E-ST-40_0860271

a. The SRS shall describe in detail the product’s relationship to other systems.

ECSS-E-ST-40_0860272

b. If the product is a component of an integrated HW–SW product, then the SRS shall:

1. summarize the essential characteristics of this larger product;
2. list the other HW or SW component the software interfaces with, and summarize the computer hardware and peripheral equipment to be used.

ECSS-E-ST-40_0860675

c. A block diagram may be presented showing the major components of the larger system or project, interconnections, and external interfaces.

<4.4> Constraints

ECSS-E-ST-40_0860274

a. The SRS shall describe any items that limit the developer’s options for building the software.

ECSS-E-ST-40_0860676

b. The SRS should provide background information and seek to justify the constraints.

<5> Requirements

<5.1> General

NOTE The following provisions apply to the software requirements listed in the SRS, as specified in <5.2> to <5.17> below.

ECSS-E-ST-40_0860276

a. Each requirement shall be uniquely identified.

ECSS-E-ST-40_0860277

b. When requirements are expressed as models, the supplier shall establish a way to assign identifiers within the model for sake of traceability.

ECSS-E-ST-40_0860278

c. The traceability information of each requirement derived from higher level documentation, to the applicable higher level requirement, shall be stated.

NOTE The documented trace can be provided automatically by tools when models are used to express requirements.

ECSS-E-ST-40_0860677

d. Requirements may be characterized, for example as essential or not, with a priority level to prepare incremental delivery, stable or not.

<5.2> Functional requirements

ECSS-E-ST-40_0860280

a. The SRS shall describe the capabilities to be provided by the software item under definition.

ECSS-E-ST-40_0860281

b. The SRS shall provide where applicable the link between the requirements and the system states and modes.

ECSS-E-ST-40_0860282

c. Functional requirement shall be grouped by subject, in accordance with the logical model organization (e.g. per controlled subsystem).

ECSS-E-ST-40_0860678

d. Each requirement definition should be organized according to the following:

1. General
2. Inputs
3. Outputs
4. Processing

ECSS-E-ST-40_0860284

e. The SRS shall describe the functional requirements related to software safety and dependability.

<5.3> Performance requirements

ECSS-E-ST-40_0860285

a. The SRS shall list any specific requirement to the specified performance of software item under definition.

<5.4> Interface requirements

ECSS-E-ST-40_0860286

a. The SRS shall list and describe (or reference in the ICD) the software item external interfaces.

ECSS-E-ST-40_0860287

b. The following interfaces shall be fully described either in the SRS itself or by reference to the ICD:

1. interfaces between the software item and other software items;
2. interfaces between the software item and hardware products;
3. interfaces requirements relating to the man–machine interaction.

ECSS-E-ST-40_0860288

c. Naming convention applicable to the data and command interface shall be also described.

ECSS-E-ST-40_0860289

d. The definition of each interface shall include at least the provided service, the description (name, type, dimension), the range and the initial value.

<5.5> Operational requirements

ECSS-E-ST-40_0860290

a. The SRS shall list any specific requirement related to the operation of the software in its intended environment.

ECSS-E-ST-40_0860679

b. The information specified in <5.5>a. should include, at least, any specified operational mode and mode transition for the software, and, in case of man–machine interaction, the intended use scenarios.

ECSS-E-ST-40_0860680

c. Diagrams may be used to show the intended operations and related modes–transitions.

<5.6> Resources requirements

ECSS-E-ST-40_0860293

a. The SRS shall describe all the resource requirements related to the software and the hardware requirements (target hardware on which the software is specified to operate), as follows:

   1. List of the requirements relevant to hardware environment in which the software is specified to operate.

   2. List of the sizing and timing requirements applicable to the software item under specification.

   3. Description of the computer software to be used with the software under specification or incorporated into the software item (e.g. operating system and software items to be reused).

   4. Description of the real time constraints to respect (e.g. time management with respect to the handling of input data before its loss of validity).

<5.7> Design requirements and implementation constraints

ECSS-E-ST-40_0860294

a. The SRS shall list any requirements driving the design of the software item under specification and any identified implementation constraint.

ECSS-E-ST-40_0860295

b. Requirements applicable to the following items shall be included:

   1. software standards (e.g. applicable coding standards, and development standards);

   2. design requirements;

   3. specific design methods to be applied to minimize the number of critical software components (see ECSS-Q-ST-80 6.2.2.4);

   4. requirements relevant to numerical accuracy management;

5. design requirements relevant to the “in–flight modification” of the software item;

6. specific design requirements to be applied if the software is specified to be designed for intended reuse;

7. specific constraints induced by reused software (e.g. COTS, free software and open source).

<5.8> Security and privacy requirements

ECSS-E-ST-40_0860296

a. The SRS shall describe any security requirement applicable to the software item.

<5.9> Portability requirements

ECSS-E-ST-40_0860297

a. The SRS shall list any portability requirement applicable to the software item.

<5.10> Software quality requirements

ECSS-E-ST-40_0860298

a. The SRS shall list any quality requirement applicable to the software item.

<5.11> Software reliability requirements

ECSS-E-ST-40_0860299

a. The SRS shall list any reliability requirement applicable to the software item.

<5.12> Software maintainability requirements

ECSS-E-ST-40_0860300

a. The SRS shall list any maintainability requirement applicable to the software item.

<5.13> Software safety requirements

ECSS-E-ST-40_0860301

a. The SRS shall list any safety requirement applicable to the software item.

<5.14> Software configuration and delivery requirements

ECSS-E-ST-40_0860302

a. The SRS shall list any requirement applicable to the selected delivery medium and any software configuration applicable to the software item.

<5.15> Data definition and database requirements

ECSS-E-ST-40_0860303

a. The SRS shall list any requirement related to specific data format or structure to be exchanged with other systems or any database requirements allowing to take into account e.g. for a flight software, the mission and product specific constraints.

<5.16> Human factors related requirements

ECSS-E-ST-40_0860304

a. The SRS shall list any requirement applicable to:

1. the personnel and to the specific software product under definition;
2. manual operations, human–equipment interactions, constraints on personnel, concentrated human attention areas and that are sensitive to human errors and training, and human factors engineering.

<5.17> Adaptation and installation requirements

ECSS-E-ST-40_0860305

a. This SRS shall list any requirement applicable to adaptation data and to specific installation.

<6> Validation requirements

ECSS-E-ST-40_0860306

a. The SRS shall describe, per each uniquely identified requirement in <5>, the validation approach.

ECSS-E-ST-40_0860307

b. A validation matrix (requirements to validation approach correlation table) shall be utilized to describe the validation approach applicable to each requirement.

<7> Traceability

ECSS-E-ST-40_0860308

a. The SRS shall report the traceability matrices

1. from the upper level specification requirements to the requirements contained in <5> (forward traceability table), and
2. from the requirements contained in <5> to the upper level applicable specification (backward traceability table).

ECSS-E-ST-40_0860309

b. In case the information in <7>a. is separately provided in the DJF, reference to this documentation shall be clearly stated.

<8> Logical model description

ECSS-E-ST-40_0860310

a. The SRS shall include a top–down description of the logical model of the software.

NOTE 1 The logical model can be the result of an iterative verification process with the customer. It also supports the requirements capture, documents and formalizes the software requirements.

NOTE 2 A logical model is a representation of the technical specification, independent of the implementation, describing the functional behaviour of the software product. The logical model is written with a formalized language and it can be possibly executable. Formal methods can be used to prove properties of the logical model itself and therefore of the technical specification. The logical model allows in particular to verify that a technical specification is complete (i.e. by checking a software requirement exists for each logical model element), and consistent (because of the model checking).

The logical model can be completed by specific feasibility analyses such as benchmarks, in order to check the technical budgets (e.g. memory size and computer throughput). In case the modelling technique allows for it, preliminary automatic code generation can be used to define the contents of the software validation test specification.

NOTE 3 If software system co-engineering activities are considered, the logical model is a refinement of the following system models: data, application function, event and failure.

ECSS-E-ST-40_0860311

b. The method used to express the logical model shall be described.

ECSS-E-ST-40_0860681

c. Diagrams, tables, data flows and explanatory text may be included.

ECSS-E-ST-40_0860682

d. The functionality at each level should be described, to enable the reader to ’walkthrough’ e.g. the model level–by–level, function–by–function, and flow–by–flow.

ECSS-E-ST-40_0860314

e. The behavioural view of the software logical model shall be also described in the SRS.

NOTE This is particularly relevant for flight software applications.

D.2.2 Special remarks

None.

Annex E (normative) Interface control document (ICD) - DRD

E.1 DRD identification

E.1.1 Requirement identification and source document

The interface control document (ICD) document is called from the normative provisions summarized in Table E-1.

Table E-1: ICD traceability to ECSS-E-ST-40 and ECSS-Q-ST-80 clauses

ECSS Standard	Clauses	DRD section
ECSS-E-ST-40	5.4.2.1 eo d	<6>
	5.4.2.1 eo g	<6>
	5.4.2.1 eo h	<5.2>
	5.4.3.5 eo a	<5.3>
	5.5.2.2 eo a	<5.3>
	5.8.3.2 eo a	<7>
eo = Expected Output		

E.1.2 Purpose and objective

The interface control document is a major constituent of the technical specification (TS). It describes all the (preliminary and update) external interfaces. This document may be part of the SRS DRD reference section <5.4>.

E.2 Expected response

E.2.1 Scope and content

<1> Introduction

ECSS-E-ST-40_0860315

a. The ICD shall contain a description of the purpose, objective, content and the reason prompting its preparation.

<2> Applicable and reference documents

ECSS-E-ST-40_0860316

a. The ICD shall list the applicable and reference documents to support the generation of the document.

<3> Terms, definitions and abbreviated terms

ECSS-E-ST-40_0860317

a. The ICD shall include any additional terms, definition or abbreviated terms used not already defined in any applicable or reference documentation.

<4> Software overview

ECSS-E-ST-40_0860684

a. The ICD may reference the software overview done in the SRS.

<5> Requirements and design

<5.1> General provisions to the requirements in the IRD

ECSS-E-ST-40_0860319

a. Each requirement shall be uniquely identified.

ECSS-E-ST-40_0860320

b. When requirements are expressed as models, the supplier shall establish a way to assign identifiers within the model for the sake of traceability.

ECSS-E-ST-40_0860321

c. The traceability information of each requirement derived from higher level documentation, to the applicable higher level requirement, shall be stated.

NOTE The documented trace can be provided automatically by tools when models are used to express requirements.

<5.2> Interface requirements

ECSS-E-ST-40_0860322

a. In case the requirements of the IRD need to be further detailed, the ICD shall list and describe the software item external interfaces.

ECSS-E-ST-40_0860323

b. The following interfaces shall be fully described:

1. interfaces between the software item and other software items;
2. interfaces between the software item and hardware products;
3. interfaces requirements relating to the man–machine interaction.

4. This can be also information about e.g.:
   - o detailed requirements on database structure,
   - o logical interface architecture,
   - o requirements on signal,
   - o communication protocols,
   - o timing requirements,
   - o required behaviour in case of error,
   - o telecommands (e.g. PUS selection, words contents),
   - o observable data,
   - o telemetry.

<5.3> Interface design

ECSS-E-ST-40_0860324

a. The ICD shall describe the external interfaces design of the software item.

ECSS-E-ST-40_0860685

b. The external interface may be expressed by models.

ECSS-E-ST-40_0860326

c. The following interfaces shall be fully described:

1. interfaces between the software item and other software items;
2. interfaces between the software item and hardware products;
3. interfaces requirements relating to the man–machine interaction.
4. This can be also information about e.g.:
   - o Physical interface architecture,
   - o Complete TM/TC plan,
   - o Design of all commands and telemetry stream,
   - o Protocol detailed implementation,
   - o specific design requirements to be applied if the software is specified to be designed for intended reuse.

ECSS-E-ST-40_0860327

d. The definition of each interface shall include at least the provided service, the description (name, type, dimension), the range and the initial value.

ECSS-E-ST-40_0860796

e. Each interface external to the software may be organized as follows:

— Data item (Name, description, unique identifier, description, source, destination, unit of measure, limit/range, accuracy, precision, frequency, rate, legality checks, data type, data representation);

— Message item;

— Communication protocol (by reference to the applicable documents).

<6> Validation requirements

ECSS-E-ST-40_0860329

a. The ICD shall describe, per each uniquely identified requirement in <5>, the validation approach.

ECSS-E-ST-40_0860330

b. A validation matrix (requirements to validation approach correlation table) shall be utilized to describe the validation approach applicable to each requirement.

<7> Traceability

ECSS-E-ST-40_0860331

a. The ICD shall report the traceability matrices

1. from the upper level specification requirements to the requirements contained in <5> (forward traceability table), and
2. from the requirements contained in <5> to the upper level applicable specification (backward traceability table).

ECSS-E-ST-40_0860332

b. In case the information in <7>a.1. is separately provided in the DJF, reference to this documentation shall be clearly stated.

E.2.2 Special remarks

None.

Annex F (normative) Software design document (SDD) - DRD

F.1 DRD identification

F.1.1 Requirement identification and source document

The software design document (SDD) is called from the normative provisions summarized in Table F-1.

Table F-1: SDD traceability to ECSS-E-ST-40 Part 1 and ECSS-Q-ST-80 clauses

ECSS Standard	Clauses	DRD section
ECSS-E-ST-40	5.4.3.1	<4.1>, <4.2>, <4.3>, <5.1>, <5.2>, <5.3>
	5.4.3.2	<4.6>, <4.7>
	5.4.3.3	<5.2>c.
	5.4.3.4	<4.3>, <5.2>e.
	5.4.3.5 eo b	<4.4>
	5.4.3.6c	<4.1>c.
	5.5.2.1a	<5.4>
	5.5.2.1b	<5.4>
	5.5.2.1c	<5.4>
	5.5.2.2 eo b	<5.5>
	5.5.2.3a eo a	<5.4>
	5.5.2.3a eo b	<5.4>
	5.5.2.3a eo c	<5.4>
	5.5.2.4	<4.7>
	5.5.2.5	<5.2>c.
	5.5.2.6	<4.7>
	5.5.2.7	<4.7>
	5.5.3.1a eo a	<5>
	5.5.3.2a eo a	<5>
	5.5.3.2b eo a	<5>
	5.8.3.3a eo a	<6>
	5.8.3.4a eo a	<6>
ECSS-Q-ST-80	7.2.2.3.b	<4.5>

F.1.2 Purpose and objective

This software design document is a constituent of the design definition file (DDF). It provides description of the software architectural design and the software detailed design. Internal interfaces design is also included in this document.

F.2 Expected response

F.2.1 Scope and content

<1> Introduction

ECSS-E-ST-40_0860333

a. The SDD shall contain a description of the purpose, objective, content and the reason prompting its preparation.

<2> Applicable and reference documents

ECSS-E-ST-40_0860334

a. The SDD shall list the applicable and reference documents to support the generation of the document.

<3> Terms, definitions and abbreviated terms

ECSS-E-ST-40_0860335

a. The SDD shall include any additional terms, definition or abbreviated terms used not already defined in any applicable or reference documentation.

<4> Software design overview

NOTE The SDD briefly introduces the system context and design and discuss the background to the project detailed as follows.

<4.1> Software static architecture

ECSS-E-ST-40_0860336

a. The SDD shall describe the architecture of the software item, as well as the main relationship between its major components.

ECSS-E-ST-40_0860337

b. The SDD shall also describe any system state or mode in which the software operates.

ECSS-E-ST-40_0860338

c. The SDD shall describe the separated mission and configuration data.

NOTE Data can be classified in the following categories:

- data resulting from the mission analysis and which thus vary from one mission to another;
- reference data which are specific to a family of software product;
- reference data which never change;
- data depending only on the specific mission requirements (e.g. calibration of sensors);
- data required for the software operation which only vary the higher level system design (in which is embedded the software) is changed.

<4.2> Software dynamic architecture

ECSS-E-ST-40_0860339

a. The SDD shall describe the design choices to cope with the real time constraints (e.g. selection and description of the computational model).

<4.3> Software behaviour

<4.4> Interfaces context

ECSS-E-ST-40_0860340

a. The SDD shall identify all the external interfaces or refer to the ICD.

ECSS-E-ST-40_0860686

b. The description in <4.4>a. should be based on system block diagram or context diagram to illustrate the relationship between this system and other systems.

<4.5> Long lifetime software

ECSS-E-ST-40_0860342

a. The SDD shall describe the design choices to cope with the long planned lifetime of the software, in particular minimum dependency on the operating system and the hardware to improve portability.

<4.6> Memory and PU budget

ECSS-E-ST-40_0860343

a. The SDD shall document and summarize the allocation of memory and processing time to the software components.

<4.7> Design standards, conventions and procedures

ECSS-E-ST-40_0860344

a. The SDD shall summarize (or refer to the SDP) the software methods adopted for the architectural and the detailed design.

NOTE A design method offers often the following characteristics:

- decomposition of the software architecture in design objects having integral parts that communicate with each other and with the outside environment;
- explicit recognition of typical activities of real-time systems (i.e. cyclic and sporadic threads, protected resources);
- integration of appropriate scheduling paradigms with the design process;
- explicit definition of the application timing requirements for each activity;
- static verification of processor allocation, schedulability and timing analysis;
- consistent code generation.

ECSS-E-ST-40_0860345

b. The following information shall be summarized:

1. software architectural design method;
2. software detailed design method;
3. code documentation standards;
4. naming conventions;
5. programming standards;
6. intended list of reuse components
7. main design trade-off.

<5> Software design

<5.1> General

ECSS-E-ST-40_0860346

a. The SDD shall describe the software architectural design.

ECSS-E-ST-40_0860347

b. The architecture structure of the software item shall be described, identifying the software components, their hierarchical relationships, any dependency and interfaces between them.

ECSS-E-ST-40_0860348

c. For flight software, the design shall reflect in flight modification requirements.

ECSS-E-ST-40_0860687

d. The structure in <5.2> to <5.5> should be used.

<5.2> Overall architecture

ECSS-E-ST-40_0860350

a. The SDD shall describe the software architectural design, from a static point of view and also, when the software to be developed has real time constraints, from a dynamic point of view, and from a behaviour point of view.

ECSS-E-ST-40_0860351

b. The software static architecture shall be summarized describing its components.

ECSS-E-ST-40_0860352

c. For real–time software, the software dynamic architecture shall be summarized describing its selected computational model.

NOTE An analysable computational model generally consists in defining:

- the types of components (objects) participating to the real-time behaviour, from which the system is constructed (e.g. active-periodic, active-sporadic, protected, passive, actors, process, blocks, drivers);
- the scheduling type (e.g. sequential or multithreaded), the scheduling model (e.g. cyclic or pre-emptive, fixed or dynamic priority based), and the analytical model (e.g. Rate Monotonic Scheduling, Deadline Monotonic Scheduling, Earliest Deadline First), under which the system is executed and its associated mechanisms;
- the means of communication between components/objects (e.g. mailboxes, entry parameters);
- the means of synchronization between components or objects (e.g. mutual exclusion, protected object entries, basic semaphores);
- If applicable, the means of distribution and internode communication (e.g. virtual nodes, Remote Procedure Call);

and (optional for non flight software):

- the means of providing timing facilities (e.g. real clock, with or without interrupt, multiple interrupting count-down, relative or absolute delays, timers time-out);
- the means of providing asynchronous transfer of control (e.g. watchdog to transfer control from anywhere to the reset sequence, software service of the underlying run-time system to cause transfer of control within the local scope of the thread).

ECSS-E-ST-40_0860688

d. The description in <5.2>c. should consist in the following information:

1. type of components participating to the real time behaviour,
2. scheduling type (e.g. single or multi–threads),
3. scheduling model (e.g. pre–emptive or not, fixed or dynamic priority based),
4. analytical model (e.g. rate monotonic scheduling, deadline monotonic scheduling),
5. Tasks identification and priorities,
6. Means of communication and synchronization,
7. Time management.

ECSS-E-ST-40_0860354

e. The software behaviour shall be described e.g. with automata or scenarios.

ECSS-E-ST-40_0860355

f. The software static, dynamic and behavioural architecture shall be described in accordance with the selected design method.

ECSS-E-ST-40_0860356

g. The SDD shall describe the error handling and fault tolerance principles (e.g. error detection, reporting, logging, and fault containment regions.).

<5.3> Software components design - General

ECSS-E-ST-40_0860357

a. The SDD shall describe:

1. The software components, constituting the software item.
2. The relationship between the software components.
3. The purpose of each software component.
4. For each software component, the development type (e.g. new development, software to be reused).
5. If the software is written for the reuse,

- its provided functionality from an external point of view, and
- its external interfaces.

6. Handling of existing reused components.

NOTE See Annex N.

ECSS-E-ST-40_0860358

b. The following shall apply to the software components specified in <5.3>a.1.:

1. Each software component is uniquely identified.
2. When components are expressed as models, the supplier establishes a way to assign identifiers within the model for sake of traceability.
3. The software requirements allocation provides for each software component;

NOTE The documented trace can be provided automatically by tools when models are used to express components.

ECSS-E-ST-40_0860689

c. The description of the components should be laid out hierarchically, in accordance with the following aspects for each component, further described in <5.4>:

- <Component identifier>
- <Type>
- <Purpose>
- <Function>
- <Subordinates>
- <Dependencies>
- <Interfaces>
- <Resources>
- <References>
- <Data>

NOTE Detailed description of the aspects for each component are describe in <5.4>.

<5.4> Software components design - Aspects of each component

<5.4.1> General

ECSS-E-ST-40_0860690

a. This part of the DRD, as well as <5.5>, may be produced as the detailed design model of a tool, if agreed with the customer.

<5.4.2> <Component identifier>

ECSS-E-ST-40_0860691

a. Each component should have a unique identifier.

ECSS-E-ST-40_0860692

b. The component should be named according to the rules of the programming language or operating system to be used.

ECSS-E-ST-40_0860693

c. A hierarchical naming scheme should be used that identifies the parent of the component (e.g. ParentName_ChildName).

<5.4.3> <Type>

ECSS-E-ST-40_0860694

a. Component type should be described by stating its logical and physical characteristics.

ECSS-E-ST-40_0860695

b. The logical characteristics should be described by stating the package, library or class that the component belongs to.

ECSS-E-ST-40_0860696

c. The physical characteristics should be described by stating the type of component, using the implementation terminology (e.g. task, subroutine, subprogram, package and file).

NOTE The contents of some components description clauses depend on the component type. For the purpose of this guide, the following categories are used: executable and non–executable.

<5.4.4> <Purpose>

ECSS-E-ST-40_0860697

a. The purpose of a component should describe its trace to the software requirements that it implements.

NOTE Backward traceability depends upon each component description explicitly referencing the requirements that justify its existence.

<5.4.5> <Function>

ECSS-E-ST-40_0860368

a. The function of a component shall be described in the software architectural design.

ECSS-E-ST-40_0860698

b. The description specified in <5.4.5>a. should be done by stating what the component does.

NOTE 1 The function description depends upon the component type. Therefore, it can be a description of the process.

NOTE 2 Process descriptions can use such techniques as structured English, precondition–postcondition specifications and state–transition diagrams.

<5.4.6> <Subordinates>

ECSS-E-ST-40_0860699

a. The subordinates of a component should be described by listing the immediate children.

NOTE 1 The subordinates of a unit are the units that are ’called by’ it. The subordinates of a database can be the files that ’compose’ it.

NOTE 2 The subordinates of an object are the objects that are ’used by’ it.

<5.4.7> <Dependencies>

ECSS-E-ST-40_0860700

a. The dependencies of a component should be described by listing the constraints upon its use by other components.

NOTE Examples are:

- Operations to take place before this component is called,
- Operations that are excluded when this operation takes place.

<5.4.8> <Interfaces>

ECSS-E-ST-40_0860372

a. Both control flow and data flow aspects of an interface shall be described for each “executable” component.

ECSS-E-ST-40_0860701

b. Data aspects of ’non executable’ components should be described.

ECSS-E-ST-40_0860702

c. The control flow to and from a component should be described in terms of how to start (e.g. subroutine call) and terminate (e.g. return) the execution of the component.

ECSS-E-ST-40_0860703

d. If the information in <5.4.8>c. is implicit in the definition of the type of component, a description need not be done.

ECSS-E-ST-40_0860704

e. If control flows take place during execution (e.g. interrupt), they should be described.

ECSS-E-ST-40_0860377

f. The data flow input to and output from each component shall be described.

ECSS-E-ST-40_0860705

g. It should be ensured that data structures:

1. are associated with the control flow (e.g. call argument list);
2. interface components through common data areas and files.

<5.4.9> <Resources>

ECSS-E-ST-40_0860706

a. The resources' needs of a component should be described by itemising what the component needs from its environment to perform its function.

NOTE 1 Items that are part of the component interface are excluded.

NOTE 2 Examples of resources' needs of a component are displays, printers and buffers.

<5.4.10> <References>

ECSS-E-ST-40_0860707

a. Explicit references should be inserted where a component description uses or implies material from another document.

<5.4.11> <Data>

ECSS-E-ST-40_0860708

a. The data internal to a component should be described.

NOTE The amount of details to be provided depends strongly on the type of the component.

ECSS-E-ST-40_0860709

b. The data structures internal to a program or subroutine should also be described.

ECSS-E-ST-40_0860383

c. Data structure definitions shall include the:

1. description of each element (e.g. name, type, dimension);
2. relationships between the elements (i.e. the structure);
3. range of possible values of each element;
4. initial values of each element.

<5.5> Internal interface design

ECSS-E-ST-40_0860384

a. The SDD shall describe the internal interfaces among the identified software components.

ECSS-E-ST-40_0860385

b. The interface data specified in a., by component, shall be organized showing the complete interfaces map, using as appropriate diagrams or matrices supporting their cross–checking.

ECSS-E-ST-40_0860386

c. For each identified internal interface, all the defined data elements shall be included.

NOTE The amount of detail to be provided depends strongly on the type of component.

ECSS-E-ST-40_0860710

d. The logical and physical data structure of files that interface major component should be postponed to the detailed design.

ECSS-E-ST-40_0860388

e. Data structure definitions shall include:

1. the description of each element (e.g. name, type, dimension);
2. the relationships between the elements (i.e. the structure);
3. the initial values of each element.

<6> Requirements to design components traceability

ECSS-E-ST-40_0860389

a. The SDD shall provide traceability matrices

1. from the software requirements to component down to the lower identified component in the software hierarchy (forward traceability) and
2. from the software components to its upper level component up to the software requirements (backward traceability).

ECSS-E-ST-40_0860390

b. In case the information in <6>a. is provided as separate documentation in the DJF, a reference to it shall be stated.

ECSS-E-ST-40_0860391

c. The SDD shall define the potential specific measures taken for critical software in the design documentation.

F.2.2 Special remarks

None.

Annex G (normative) Software release document (SRelD) - DRD

G.1 DRD identification

G.1.1 Requirement identification and source document

The software release document (SRelD) is called from the normative provisions summarized in Table G-1.

Table G-1: SRelD traceability to ECSS-E-ST-40 and ECSS-QST--80 clauses

ECSS Standard	Clauses	DRD section
ECSS-E-ST-40	5.7.2.1a eo b	All
ECSS-Q-ST-80	6.2.4.3 eo b	All
eo = Expected Output		

G.1.2 Purpose and objective

The SRelD is a constituent of the DDF. Its purpose is to describe a given software version including known problems, limitations or restrictions with respect to its approved baseline.

G.2 Expected response

G.2.1 Scope and content

<1> Introduction

ECSS-E-ST-40_0860392

a. The SRelD shall contain a description of the purpose, objective, content and the reason prompting its preparation.

<2> Applicable and reference documents

ECSS-E-ST-40_0860393

a. The SRelD shall list the applicable and reference documents to support the generation of the document.

<3> Terms, definitions and abbreviated terms

ECSS-E-ST-40_0860394

a. The SRelD shall include any additional terms, definition or abbreviated terms used not already defined in any applicable or reference documentation.

<4> Software release overview

ECSS-E-ST-40_0860395

a. The SRelD shall contain a brief description of the information to be associated with a software release, including:

1. reference of the corresponding SCF,
2. version of the delivered software configuration item,
3. status of SPRs, SCRs and SW&D related to the software configuration item, and
4. advice for use of the software configuration item.

NOTE The software release document is a subset of the software configuration file that describes a new version by comparison with "reference" or the previous one. It is used for the delivery of a new version of a software configuration item to a customer.

<5> Status of the software configuration item

<5.1> Evolution since previous version

ECSS-E-ST-40_0860396

a. The SRelD shall

1. summarize the main information on the software configuration item, and
2. describe the changes implemented since previous version.

<5.2> Known problems or limitations

ECSS-E-ST-40_0860397

a. The SRelD shall list all the unsolved SPR and approved SW&D related to the version of the software configuration item.

<6> Advice for use of the software configuration item

ECSS-E-ST-40_0860398

a. The SRelD shall provide advice for the use of this version of the software configuration item.

NOTE For example: Potential problems, and compatibility with other configuration items).

<7> On–going changes

ECSS-E-ST-40_0860399

a. The SRelD shall provide information on planned evolution of the software configuration item.

G.2.2 Special remarks

None.

Annex H (normative) Software user manual (SUM) - DRD

H.1 DRD identification

H.1.1 Requirement identification and source document

The software user manual (SUM) is called from the normative provisions summarized in Table H-1.

Table H-1: SUM traceability to ECSS-E-ST-40 and ECSS-Q-ST-80 clauses

ECSS Standard	Clauses	DRD section
ECSS-E-ST-40	5.5.2.8	All
	5.6.3.3	All
	5.6.4.3	All

H.1.2 Purpose and objective

The software user manual is a constituent of the design definition file (DDF). Its purpose is to provide instructions for the users of the software.

The user manual can be split in different documents addressing the needs of the different roles involved (e.g. operator, administrator, security officer, etc.).

For flight software, the relevant parts of the SUM are a contribution to the flight operation manual (FOM).

H.2 Expected response

H.2.1 Scope and content

<1> Introduction

ECSS-E-ST-40_0860400

a. The SUM shall contain a description of the purpose, objective, content and the reason prompting its preparation.

<2> Applicable and reference documents

ECSS-E-ST-40_0860401

a. The SUM shall list the applicable and reference documents to support the generation of the document.

<3> Terms, definitions and abbreviated terms

ECSS-E-ST-40_0860402

a. The SUM shall include any additional terms, definition or abbreviated terms used not already defined in any applicable or reference documentation.

<4> Conventions

ECSS-E-ST-40_0860403

a. The SUM shall summarise symbols, stylistics conventions, and command syntax conventions used in the document.

NOTE An example of stylistic conventions is using boldface and courier font to distinguish user input. Examples of syntax conventions are the rules for combining commands, keywords and parameters.

<5> Purpose of the Software

ECSS-E-ST-40_0860404

a. The SUM shall include a description of the intended uses of the software, in terms of capabilities, operating improvements, and benefits expected from its use.

<6> External view of the software

ECSS-E-ST-40_0860405

a. The SUM shall identify the software files, including databases and data files, which are necessary for the software to operate, including security and privacy considerations for each file and identification of the software necessary to continue or resume operation in case of an emergency.

<7> Operations environment

<7.1> General

ECSS-E-ST-40_0860406

a. The SUM shall describe the configuration hardware and software of the environment, the identification of all system components and the hardware, software, manual operations, and other resources needed for a user to install and run the software.

<7.2> Hardware configuration

ECSS-E-ST-40_0860407

a. The SUM shall describe through a block diagram the principal hardware parts and configuration of FPGAs of the system, the communications equipment, the computer equipment, including amount of memory needed, amount of auxiliary storage needed, and peripheral equipment such as printers and other input/output devices;

<7.3> Software configuration

ECSS-E-ST-40_0860408

a. The SUM shall describe through a block diagram the principal software parts of the system, including other software, such as operating systems, utilities, other supporting systems, and other facilities, equipment, or resources.

<7.4> Operational constraints

ECSS-E-ST-40_0860409

a. The SUM shall explain the what the user can do with the software in various states and modes of operation, including emergency degraded modes.

<8> Operations basics

ECSS-E-ST-40_0860410

a. The SUM shall define the operational tasks, identifying their sequence and hierarchy, the roles and the staffing, the standard daily operations and the contingency operations.

<9> Operations manual

<9.1> General

ECSS-E-ST-40_0860411

a. The SUM shall contain the operational organisation, a reference schedule for each operational profile, the list of all the elementary operations to be carried out at the site, what to do in order to operate the site, the personnel responsible to do it and when.

<9.2> Set-up and initialisation

ECSS-E-ST-40_0860412

a. The SUM shall describe any procedures to be performed by the user in order to be identified or authorised to access or install software on the equipment, to perform the installation, to configure the software, to delete or overwrite former files or data, and to enter parameters for software operation.

<9.3> Getting started

ECSS-E-ST-40_0860413

a. The SUM shall include the step-by-step procedures for beginning work, including any options available, and a check-list for problem determination.

<9.4> Access

ECSS-E-ST-40_0860414

a. The SUM shall give an overview of the access and security features of the software that are visible to the user, and in particular:

— How and from whom to obtain a password

— How to add, delete, or change passwords under user control

— Security and privacy considerations pertaining to the storage and marking of output reports and other media that the user can generate

<9.5> Normal operations

ECSS-E-ST-40_0860415

a. The SUM shall identify the normal operations, to be performed by the user, for the use of software (function, menu, transaction, or other process being described), including description and options of menus, graphical icons, data entry forms, user inputs, inputs from other software or hardware affecting the software's interface with the user, outputs, diagnostic or error messages or alarms.

<9.6> Normal termination

ECSS-E-ST-40_0860416

a. The SUM shall describe how the user can cease or interrupt use of the software and how to determine whether normal termination or cessation has occurred.

<9.7> Error conditions

ECSS-E-ST-40_0860417

a. The SUM shall describe the common error conditions that can occur as a result of executing the function, and how to detect that the error has occurred.

<9.8> Recover runs

ECSS-E-ST-40_0860418

a. The SUM shall include the detailed procedures for restart or recovery from errors or malfunctions occurring during processing and for ensuring continuity of operations in the event of emergencies.

<9.9> Secure installation and operation

ECSS-E-ST-40_0860797

a. The SUM shall include the information regarding secure installation and configuration parameters for software, system and network; and secure operation instructions and procedures.

<10> Reference manual

<10.1> Introduction

ECSS-E-ST-40_0860419

a. The SUM shall provide or reference, a quick-reference card or page for using the software, which summarises frequently used function keys, control sequences, formats, commands, or other aspects of software use.

<10.2> Help method

ECSS-E-ST-40_0860420

a. The SUM shall include the help information about the actions to be performed by the user, how to invoke the function, possible errors, how to resolve them and what results to expect.

<10.3> Screen definitions and operations

ECSS-E-ST-40_0860421

a. The SUM shall include the description of the dimensions and capabilities of the visual display screen.

<10.4> Commands and operations

ECSS-E-ST-40_0860422

a. The SUM shall include a guide to the command language used, operations and functions.

<10.5> Error messages

ECSS-E-ST-40_0860423

a. The SUM shall list all error messages, diagnostic messages, and information messages that can occur while accomplishing any of the user’s functions, including the meaning of each message and the action to be taken after each such message.

<11> Tutorial

<11.1> Introduction

ECSS-E-ST-40_0860424

a. The SUM shall describe how to use the software and what the software does, combining tutorials and reference information for both novices and experts.

<11.2> Getting started

ECSS-E-ST-40_0860425

a. The SUM shall include a welcoming introduction to the software.

<11.3> Using the software on a typical task

ECSS-E-ST-40_0860426

a. The SUM shall describe a typical use case of the software, using graphical pictures and diagrams to demonstrate the actions performed by the user.

<12> Analytical Index

ECSS-E-ST-40_0860427

a. The SUM, if more than 40 pages, shall include an index containing a systematic list of topics from the user's point of view, the major synonyms and variants (especially if these are well known to users but are not employed in the operational manual for technical reasons), pointing to topics in the body of the manual by:

— page number,

— section number,

— illustration number,

— primary index entry (one level of reference only).

NOTE Index entries usefully contain auxiliary information, especially cross-references to contrasting or related terms. For example, the entry for INSERT says 'see also DELETE'. Indexes are made particularly helpful if attention is drawn primarily to important keywords, and to important locations in the body of the manual. This is achieved by highlighting such entries, and by grouping minor entries under major headings. Indexes do not contain more than two levels of entry. If a single index points to different kinds of location, such as pages and illustration numbers, these are unambiguously distinguished, (e.g. Page 35, Figure 7), since the use of highlighting (35, 7) is not for instance sufficient to prevent confusion in this case.

H.2.2 Special remarks

None.

Annex I (normative) Software verification plan (SVerP) - DRD

I.1 DRD identification

I.1.1 Requirement identification and source document

The software verification plan (SVerP) is called from the normative provisions summarized in Table I-1.

Table I-1: SVerP traceability to ECSS-E-ST-40 and ECSS-Q-ST-80 clauses

ECSS Standard	Clauses	DRD section
ECSS-E-ST-40	5.8.2.1a	<4>
	5.8.2.1b	<4>
	5.8.2.1c	<6>
	5.8.2.1d	<4>
	5.8.2.1e	<4>
ECSS-Q-ST-80	6.2.6.1	<6.3>
eo = Expected Output		

I.1.2 Purpose and objective

The software verification plan is a constituent of the design justification file (DJF). Its purpose of the software verification plan is to describe the approach and the organization aspects to implement the software verification activities. Based upon the list of verification tasks, the verification plan address the following items:

- the life cycle activities and software products subject to verification;
- the required verification tasks for each life cycle activity, software product, related resources, responsibilities, and schedule;
- the procedures for forwarding verification reports to the customer and other involved organizations.

I.2 Expected response

I.2.1 Scope and content

<1> Introduction

ECSS-E-ST-40_0860428

a. The SVerP shall contain a description of the purpose, objective, content and the reason prompting its preparation.

<2> Applicable and reference documents

ECSS-E-ST-40_0860429

a. The SVerP shall list the applicable and reference documents to support the generation of the document.

<3> Terms, definitions and abbreviated terms

ECSS-E-ST-40_0860430

a. The SVerP shall include any additional terms, definition or abbreviated terms used not already defined in any applicable or reference documentation.

<4> Software verification process overview

<4.1> General

ECSS-E-ST-40_0860431

a. The SVerP shall describe the approach to be utilized to implement the verification process throughout the software life cycle, the verification effort, and the level of independence for the verification tasks, as follows:

NOTE 1 It is important to check the applicability of ECSS-Q-ST-80 clause 5.3.1 (management of risks), 6.2.2 (software dependability and safety) and 6.2.6.13 (independent software verification and validation).

NOTE 2 The verification effort is sized according to:

- the potential for an undetected error in a system or software requirement to cause death or personal injury, mission failure, or financial or catastrophic equipment loss or damage;
- the maturity of and risks associated with the software technology to be used;
- availability of funds and resources.

<4.2> Organization

ECSS-E-ST-40_0860432

a. The SVerP shall describe the organization of the documentation review, proofs, and tracing activities.

ECSS-E-ST-40_0860433

b. The following topics that shall be included:

1. roles;
2. reporting channels;
3. levels of authority for resolving problems;
4. organization relationships;
5. level of required and implemented independence.

<4.3> Master schedule

ECSS-E-ST-40_0860434

a. A reference to the master schedule given in the software development plan shall be done.

ECSS-E-ST-40_0860435

b. This SVerP shall describe the schedule for the planned verification activities.

<4.4> Resource summary

ECSS-E-ST-40_0860436

a. The SVerP shall summarize the resources to be used to perform the verification activities such as staff, hardware and software tools.

<4.5> Responsibilities

ECSS-E-ST-40_0860437

a. The SVerP shall describe the specific responsibilities.

<4.6> Identification of risks and level of independence.

ECSS-E-ST-40_0860438

a. The SVerP shall state (or refer to the SDP) the risks and level of independence.

<4.7> Tools, techniques and methods

ECSS-E-ST-40_0860439

a. The SVerP shall describe the software tools, techniques and methods used to execute the verification tasks throughout the software life cycle.

<5> Control procedures for verification process

ECSS-E-ST-40_0860440

a. The SVerP shall contain information (or reference to) about applicable management procedures concerning the following aspects:

1. problem reporting and resolution;
2. deviation and waiver policy;
3. control procedures.

<6> Verification activities

<6.1> General

ECSS-E-ST-40_0860441

a. The SVerP shall address the verification activities of each software item.

ECSS-E-ST-40_0860442

b. The SVerP shall address separately the activities to be performed for manually and automatically generated code.

<6.2> Software process verification

ECSS-E-ST-40_0860443

a. For each software process verification, the SVerP shall list:

1. the verification activities to be performed and how they are performed.
2. the required inputs to achieve the verification activities.
3. the intermediate and final outputs documenting the performed verification activities.
4. the methodologies, tools and facilities utilized to accomplish the verification activities.

NOTE Examples of input and output are:

- for software requirements (RB and TS) and architecture engineering:

  - input: draft SRS, draft software architectural design

  - output: software verification requirements report, software architectural design to requirements traceability

- for software design and implementation engineering:

  - input: software components design, code, software user manual, software integration test plan

  - output: software code verification report, evaluation of software validation testing specification

- for software delivery and acceptance:
  - input: software validation specification with respect to the requirements baseline, software acceptance testing documentation
  - output: software acceptance test report, software acceptance data package, problem reports, software release document, software configuration file
- for software validation:
  - input: software validation specification with respect to the requirements baseline
  - output: software validation testing specifications

<6.3> Software quality requirements verification (as per ECSS-Q-ST-80 clause 6.2.6.1)

<6.3.1> Activities

ECSS-E-ST-40_0860444

a. The SVerP shall list the verification activities to be performed and how these are accomplished.

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.

<6.3.2> Inputs

ECSS-E-ST-40_0860445

a. The SVerP shall list the required inputs to accomplish the verification activities.

<6.3.3> Outputs

ECSS-E-ST-40_0860446

a. The SVerP shall list the intermediate and final outputs documenting the performed verification activities.

<6.3.4> Methodology, tools and facilities

ECSS-E-ST-40_0860447

a. The SVerP shall describe the methodologies, tools and facilities utilized to accomplish the software quality requirements verification activities.

I.2.2 Special remarks

None.

Annex J (normative) Software validation plan (SValP) - DRD

J.1 DRD identification

J.1.1 Requirement identification and source document

The software validation plan (SValP) is called from the normative provisions summarized in Table J-1.

Table J-1: SValP traceability to ECSS-E-ST-40 and ECSS-Q-ST-80 clauses

ECSS Standard	Clauses	DRD section
ECSS-E-ST-40	5.6.2.1a	<4>, <6>
	5.6.2.1b	<4.6>, <5>, <7>
	5.6.2.1c	<4>
	5.8.3.9a (TS + RB)	<9>
ECSS-Q-ST-80	6.2.8.2	<4.1>c.
	6.2.8.7	<4.1>c.
	6.3.5.22	<4>
	6.3.5.23	<4.4>
	6.3.5.24	<4.6>
	6.3.5.25	<5>
	6.3.5.29	<6>

J.1.2 Purpose and objective

The software validation plan is a constituent of the design justification file (DJF). Its purpose is to provide the definition of organizational aspects and management approach to the implementation of the validation tasks.

The objective of the software validation plan is to describe the approach to the implementation of the validation process for a software product.

J.2 Expected response

J.2.1 Scope and content

<1> Introduction

ECSS-E-ST-40_0860448

a. The SValP shall contain a description of the purpose, objective, content and the reason prompting its preparation.

<2> Applicable and reference documents

ECSS-E-ST-40_0860449

a. The SValP shall list the applicable and reference documents to support the generation of the document.

<3> Terms, definitions and abbreviated terms

ECSS-E-ST-40_0860450

a. The SValP shall include any additional terms, definition or abbreviated terms used not already defined in any applicable or reference documentation.

<4> Software validation process planning

<4.1> General

ECSS-E-ST-40_0860451

a. The SValP shall describe the approach to be utilized to implement the validation process, the required effort, and the level of required independence for the validation tasks.

ECSS-E-ST-40_0860452

b. The SValP shall also address, if it is applicable to the software validation campaign against the requirements baseline, to the software validation campaign against the technical specification, or, for both.

ECSS-E-ST-40_0860453

c. The SValP shall address separately the activities to be performed for manually and automatically generated code.

ECSS-E-ST-40_0860454

d. The SValP shall include the validation of the quality requirements.

<4.2> Organization

ECSS-E-ST-40_0860455

a. The SValP shall describe the organization of the validation activities.

ECSS-E-ST-40_0860456

b. Topics that shall be included are:

1. organizational structure;
2. relationships to the other activities such as project management, development, configuration management and product assurance;
3. level of required and implemented independence in validation activities execution.

<4.3> Schedule

ECSS-E-ST-40_0860457

a. A reference to the master schedule given in the software development plan shall be included.

ECSS-E-ST-40_0860458

b. The SValP shall describe the schedule for the planned validation activities. In particular, test milestones identified in the software project schedule and all item delivery events.

ECSS-E-ST-40_0860459

c. The SValP shall describe:

1. the schedule for each testing task and test milestone;
2. the period of use for the test facilities.

<4.4> Resource summary

ECSS-E-ST-40_0860460

a. The SValP shall summarize the resources needed to perform the validation activities such as staff, hardware, software tools, testing data and support software (simulators).

<4.5> Responsibilities

ECSS-E-ST-40_0860461

a. The SValP shall describe the specific responsibilities associated with the roles described in <4.2> above.

ECSS-E-ST-40_0860462

b. In particular, the SValP shall state the groups responsible for managing, designing, preparing, executing witnessing and checking tests results.

NOTE Groups can include developers, operational staff, user representatives, technical support staff and product assurance staff.

<4.6> Tools, techniques and methods

ECSS-E-ST-40_0860463

a. The SValP shall describe the software tools, techniques and methods used for validation activities as well as the needed hardware facilities and, testing data, support software (simulators).

ECSS-E-ST-40_0860464

b. The SValP shall describe the validation facility in terms of:

1. level of representativeness of the physical and functional environment, including the processor and the real-time representativeness;
2. software or hardware in the loop;
3. open or closed loop capability for functional and performance testing;
4. debugging and observability capability;
5. For real-time software, the constraints on the test execution such as interdiction of code instrumentation, and test method (e.g. referring to measurement techniques and tools) associated to performance or safety requirements.

<4.7> Personnel requirements

ECSS-E-ST-40_0860465

a. The SValP shall describe any requirement for software validation personnel (level of independence) and any necessary training needs.

<4.8> Risks

ECSS-E-ST-40_0860466

a. The SValP shall state (or refer to the SDP) all the identified risks to the software validation campaign.

ECSS-E-ST-40_0860467

b. Contingency plans shall be also included.

<5> Software validation tasks identification

ECSS-E-ST-40_0860468

a. The SValP shall describe the software validation tasks to be performed for the identified software items.

ECSS-E-ST-40_0860469

b. The SValP shall list which are the tasks and the items under tests, as well as the criteria to be utilized for the testing activities on the test items associated with the plan.

ECSS-E-ST-40_0860470

c. The SValP shall list the testing activities to be repeated when testing is resumed.

ECSS-E-ST-40_0860471

d. The SValP shall describe for each validation tasks the inputs, the outputs as well as the resources to be used for each task.

ECSS-E-ST-40_0860472

e. The detailed information and the data for the testing procedures shall be provided in the software validation testing specifications.

<6> Software validation approach

ECSS-E-ST-40_0860473

a. The SValP shall describe the overall requirements applicable to the software validation testing activities, providing for definition of overall requirements, guidelines on the kinds of tests to be executed.

ECSS-E-ST-40_0860474

b. The SValP shall describe the selected approach to accomplish validation of those software specification requirements to be validated by inspection and analysis or review of design.

ECSS-E-ST-40_0860475

c. The SValP shall define the regression testing strategy.

<7> Software validation testing facilities

ECSS-E-ST-40_0860476

a. This SValP shall describe the test environment to execute the software validation testing activity and the non–testing validation activities whose approach is defined by this plan.

ECSS-E-ST-40_0860477

b. The SValP shall describe the configuration of selected validation facilities in terms of software (e.g. tools and programs, and simulation), hardware (e.g. platforms and target computer), test equipment (e.g. bus analyser), communications networks, testing data and support software (e.g. simulators).

NOTE Reference to other documentation describing the facility can be done.

ECSS-E-ST-40_0860478

c. If the validation testing against the requirements baseline and the validation testing against the technical specification use different environments, this shall be clearly stated and described.

<8> Control procedures for software validation process

ECSS-E-ST-40_0860479

a. The SValP shall contain information (or reference to) about applicable management procedures concerning the following aspects:

1. problem reporting and resolution;
2. deviation and waiver policy;
3. configuration control and management.

<9> Complement of validation at system level

ECSS-E-ST-40_0860480

a. The SVS w.r.t. TS or RB shall list the requirements of the TS or RB that cannot be tested in the validation environment and need the full real system to be tested, therefore including the customer support.

J.2.2 Special remarks

None.

Annex K (normative) Software [unit/integration] test plan (SUITP) - DRD

K.1 DRD identification

K.1.1 Requirement identification and source document

The software [unit/integration] test plan (SUITP) is called from the normative provisions summarized in Table K-1.

Table K-1: SUITP traceability to ECSS-E-ST-40 and ECSS-Q-ST-80 clauses

ECSS Standard	Clauses	DRD section
ECSS-E-ST-40	5.4.3.8 (IT)	<5>, <6>, <7>
	5.5.4.1 (IT)	<8>, <9>, <10>, <11>
	5.5.2.9 (UT)	<5>, <6>, <7>, <8>, <9>
	5.5.3.2a eo b (UT)	<10>, <11>
ECSS-Q-ST-80	6.2.8.2	<7.6>a.
	6.2.8.7	<7.6>a.
	6.3.5.22	<5>
	6.3.5.23	<5.3>
	6.3.5.24	<5.5>
	6.3.5.25	<9.2>, <9.2.7>, <10>
eo = Expected Output		

K.1.2 Purpose and objective

The software unit test plan and the software integration test plan are constituents of the design justification file.

The purpose of this DRD is to describe the tests plans, and is utilized for the following documents:

- the software unit test plan;
- the software integration test plan.

It provides a unique template for unit and integration testing, to be instantiated for the software test plans specified in the document requirement list, either for a software unit test plan, or for a software integration test plan. The acronym

SUITP is used to designate either the software unit test plan, or the software integration test plan.

K.2 Expected response

K.2.1 Scope and content

<1> Introduction

ECSS-E-ST-40_0860481

a. The SUITP shall contain a description of the purpose, objective, content and the reason prompting its preparation.

<2> Applicable and reference documents

ECSS-E-ST-40_0860482

a. The SUITP shall list the applicable and reference documents to support the generation of the document.

<3> Terms, definitions and abbreviated terms

ECSS-E-ST-40_0860483

a. The SUITP shall include any additional terms, definition or abbreviated terms used not already defined in any applicable or reference documentation.

<4> Software overview

ECSS-E-ST-40_0860484

a. The SUITP shall contain a brief description of the software under test and its context: a summary of its functionality, its configuration, its operational environment and its external interfaces.

NOTE Reference to technical documentation can be done.

<5> Software unit testing and software integration testing

NOTE The SUITP describes the responsibility and schedule information for the software unit testing and integration testing, detailed as follows.

<5.1> Organization

ECSS-E-ST-40_0860485

a. The SUITP shall describe the organization of software unit testing and integration testing activities.

ECSS-E-ST-40_0860711

b. The following topics should be included:

1. roles,
2. reporting channels,
3. levels of authority for resolving problems,
4. relationships to the other activities such as project management, development, configuration management and product assurance.

<5.2> Master schedule

ECSS-E-ST-40_0860487

a. The SUITP shall describe the schedule for the software unit testing and integration testing activities, in particular, test milestones identified in the software project schedule and all item delivery events.

ECSS-E-ST-40_0860712

b. The SUITP should include:

1. a reference to the master schedule given in the software development plan,
2. any additional test milestones and state the time required for each testing task,
3. the schedule for each testing task and test milestone,
4. the period of use for the test facilities.

<5.3> Resource summary

ECSS-E-ST-40_0860489

a. The SUITP shall summarize the resources needed to perform the software unit testing / integration testing activities such as staff, hardware and software tools.

<5.4> Responsibilities

ECSS-E-ST-40_0860490

a. The SUITP shall describe the specific responsibilities associated with the roles described in a.

ECSS-E-ST-40_0860713

b. The responsibilities specified in <5.4>a. should be described by identifying the groups responsible for managing, designing, preparing, executing the tests.

NOTE Groups can include developers, technical support staff, and product assurance staff.

<5.5> Tools, techniques and methods

ECSS-E-ST-40_0860492

a. The SUITP shall describe the hardware platforms, software tools, techniques and methods used for software unit testing and integration testing activities.

NOTE Some specific testing technique examples, relevant to security, would be: fuzzing, symbolic execution.

<5.6> Personnel and personnel training requirements

ECSS-E-ST-40_0860493

a. The SUITP shall list any requirement for software unit testing and integration testing personnel and their training needs.

<5.7> Risks and contingencies

ECSS-E-ST-40_0860494

a. The SUITP shall describe (or refer to the SDP) risks to the software unit testing and integration testing campaign.

ECSS-E-ST-40_0860714

b. Contingency plans should be included.

<6> Control procedures for software unit testing / integration testing

ECSS-E-ST-40_0860496

a. The SUITP shall contain information (or reference to) about applicable management procedures concerning the following aspects:

1. problem reporting and resolution;
2. deviation and waiver policy;
3. control procedures.

<7> Software unit testing and integration testing approach

NOTE The SUITP describes the approach to be utilized for the software unit testing and integration testing, detailed as follows.

<7.1> Unit/integration testing strategy

ECSS-E-ST-40_0860497

a. The SUITP shall describe the software integration strategy

<7.2> Tasks and items under test

ECSS-E-ST-40_0860498

a. The SUITP shall describe which are the tasks and the items under tests, as well as criteria to be utilized.

<7.3> Features to be tested

ECSS-E-ST-40_0860499

a. The SUITP shall describe all the features to be tested, making references to the applicable documentation.

<7.4> Features not to be tested

ECSS-E-ST-40_0860500

a. The SUITP shall describe all the features and significant combinations not to be tested.

<7.5> Test pass - fail criteria

ECSS-E-ST-40_0860501

a. The SUITP shall describe the general criteria to be used to determine whether or not test are passed.

<7.6> Manually and automatically generated code

ECSS-E-ST-40_0860502

a. The SUITP shall address separately the activities to be performed for manually and automatically generated code, although they have the same objective (ECSS-Q-ST-80 clause 6.2.8.2 and 6.2.8.7).

<8> Software unit test / integration test design

<8.1> General

ECSS-E-ST-40_0860503

a. The SUITP shall provide the definition of unit and integration test design.

ECSS-E-ST-40_0860504

b. For each identified test design, the SUITP shall provide the information given in <8.2>

NOTE This can be simplified in the software unit test plan.

<8.2> Organization of each identified test design

NOTE The SUITP provides the definition of each unit test and integration test design, detailed as follows.

<8.2.1> Test design identifier

ECSS-E-ST-40_0860505

a. The SUITP shall identify each test design uniquely.

ECSS-E-ST-40_0860506

b. The SUITP shall briefly describe the test design.

<8.2.2> Features to be tested

ECSS-E-ST-40_0860507

a. The SUITP shall list the test items and describe the features to be tested.

ECSS-E-ST-40_0860508

b. Reference to appropriate documentation shall be made and traceability information shall be provided.

<8.2.3> Approach refinements

ECSS-E-ST-40_0860509

a. The SUITP shall describe the test approach implemented for the specific test design and the specific test class.

ECSS-E-ST-40_0860510

b. The description specified in a. shall provide the rationale for the test case selection and grouping into test procedures.

ECSS-E-ST-40_0860511

c. The method for analysing test results shall be identified (e.g. compare with expected output).

ECSS-E-ST-40_0860512

d. Configuration of the facility (both hardware and software) to be used to execute the identified test shall be described.

<8.2.4> Test case identifier

ECSS-E-ST-40_0860513

a. The SUITP shall list the test cases associated with the test design and provide a summary description of each ones.

<9> Software unit and integration test case specification

<9.1> General

ECSS-E-ST-40_0860514

a. The SUITP shall provide an identification of software unit test and integration test cases.

ECSS-E-ST-40_0860515

b. For each identified test case, the SUITP shall provide the information given in <9.2>.

NOTE Each test case can be described through one or several description sheets.

<9.2> Organization of each identified test case

NOTE The SUITP provides the definition of each unit and integration test case, detailed as follows.

<9.2.1> Test case identifier

ECSS-E-ST-40_0860516

a. The SUITP shall identify the test case uniquely.

ECSS-E-ST-40_0860517

b. A short description of the test case purpose shall be provided.

<9.2.2> Test items

ECSS-E-ST-40_0860518

a. The SUITP shall list the test items.

ECSS-E-ST-40_0860519

b. Reference to appropriate documentation shall be performed and traceability information shall be provided.

<9.2.3> Inputs specification

ECSS-E-ST-40_0860520

a. The SUITP shall describe the inputs to execute the test case.

<9.2.4> Outputs specification

ECSS-E-ST-40_0860521

a. This SUITP shall describe the expected outputs.

<9.2.5> Test pass - fail criteria

ECSS-E-ST-40_0860522

a. The SUITP shall list the criteria to decide whether the test has passed or failed.

<9.2.6> Environmental needs

ECSS-E-ST-40_0860523

a. The SUITP shall describe:

1. the exact configuration and the set up of the facility used to execute the test case as well as the utilization of any special test equipment (e.g. bus analyser);

2. the configuration of the software utilized to support the test conduction (e.g. identification of the simulation configuration);

<9.2.7> Special procedural constraints (ECSS-Q-ST-80 clause 6.3.5.25)

ECSS-E-ST-40_0860524

a. The SUITP shall describe any special constraints on the used test procedures.

<9.2.8> Interfaces dependencies

ECSS-E-ST-40_0860525

a. The SUITP shall describe all the test cases to be executed before this test case.

<9.2.9> Test script

ECSS-E-ST-40_0860526

a. The SUITP shall describe all the test script used to execute the test case.

NOTE The test scripts can be collected in an appendix.

<10> Software unit and integration test procedures

<10.1> General

ECSS-E-ST-40_0860527

a. The SUITP shall provide an identification of software unit and integration test procedures.

ECSS-E-ST-40_0860528

b. For each identified test procedure, the SUITP shall provide the information given in <10.2>.

<10.2> Organization of each identified test procedure

NOTE The SUITP provides the definition of each unit and integration test procedure, detailed as follows.

<10.2.1> Test procedures identifier

ECSS-E-ST-40_0860529

a. The SUITP shall include a statement specifying the test procedure uniquely.

<10.2.2> Purpose

ECSS-E-ST-40_0860530

a. The SUITP shall describe the purpose of this procedure.

ECSS-E-ST-40_0860531

b. A reference to each test case implemented by the test procedure shall be given.

<10.2.3> Procedure steps

ECSS-E-ST-40_0860532

a. The SUITP shall describe every step of the procedure execution:

1. log: describe any special methods or format for logging the results of test execution, the incidents observed, and any other event pertinent to this test;
2. set up: describe the sequence of actions to set up the procedure execution;
3. start: describe the actions to begin the procedure execution;
4. proceed: describe the actions during the procedure execution;
5. test result acquisition: describe how the test measurements is made;
6. shut down: describe the action to suspend testing when interruption is forced by unscheduled events;
7. restart: identify any procedural restart points and describe the actions to restart the procedure at each of these points;
8. wrap up: describe the actions to terminate testing.

<11> Software test plan additional information

ECSS-E-ST-40_0860533

a. The following additional information shall be provided:

1. test procedures to test cases traceability matrix;
2. test cases to test procedures traceability matrix;
3. test scripts;
4. detailed test procedures.

NOTE 1 This information can be given in separate appendices.

NOTE 2 One test design uses one or more test cases.

NOTE 3 One test procedure execute one or more test cases.

K.2.2 Special remarks

None.

Annex L (normative) Software validation specification (SVS) - DRD

L.1 DRD identification

L.1.1 Requirement identification and source document

The software validation specification (SVS) is called from the normative provisions summarized in Table L-1.

Table L-1: SVS traceability to ECSS-E-ST-40 and ECSS-Q-ST-80 clauses

ECSS Standard	Clauses	DRD section
ECSS-E-ST-40	5.6.3.1a (TS)	<4>, <5>, <6>, <7>, <8>, <11>, <10>
	5.6.3.1b (TS)	<4>, <5>, <6>, <7>, <8>, <11>, <10>
	5.6.3.1c	<5>, <9>
	5.6.4.1a (RB)	<4>, <5>, <6>, <7>, <8>, <11>, <10>
	5.6.4.1b (RB)	<4>, <5>, <6>, <7>, <8>, <11>, <10>
	5.6.4.1c	<5>, <9>
	5.8.3.8a eo a (RB)	<11>
	5.8.3.8a eo b (TS)	<11>
ECSS-Q-ST-80	6.2.8.2	<5>
	6.2.8.7	<5>
	6.3.5.25	<7.2>, <7.2.6>, <8>
	6.3.5.29	<6>
	6.3.5.32	<5>
eo = Expected Output		

L.1.2 Purpose and objective

The software validation specification with respect to the technical specification and the software validation specification with respect to the requirements baseline are constituents of the design justification file.

The purpose of this DRD is to describe the testing, analysis, inspection and review of design specifications, and is used to document

- the software validation specification with respect to the technical specification (TS), and
- the software validation specification with respect to the requirements baseline (RB).

It provides a unique template for the software validation specification document, to be instantiated for, either the technical specification, or the requirements baseline. The acronym SVS w.r.t. TS is used to designate the software validation specification with respect to the technical specification whilst SVS w.r.t. RB is used to designate the software validation specification with respect to the requirements baseline.

L.2 Expected response

L.2.1 Scope and content

<1> Introduction

ECSS-E-ST-40_0860534

a. The SVS w.r.t. TS or RB shall contain a description of the purpose, objective, content and the reason prompting its preparation.

<2> Applicable and reference documents

ECSS-E-ST-40_0860535

a. The SVS w.r.t. TS or RB shall list the applicable and reference documents to support the generation of the document.

<3> Terms, definitions and abbreviated terms

ECSS-E-ST-40_0860536

a. The SVS w.r.t. TS or RB shall include any additional terms, definition or abbreviated terms used not already defined in any applicable or reference documentation.

<4> Software overview

ECSS-E-ST-40_0860537

a. The SVS w.r.t. TS or RB shall contain a brief description of the software under test and its context: a summary of its functionality, its configuration, its operational environment and its external interfaces.

NOTE Reference to technical documentation can be done.

<5> Software validation specification task identification

NOTE The SVS w.r.t. TS or RB describes the approach to be utilized for the software validation specification, detailed as follows.

<5.1> Task and criteria

ECSS-E-ST-40_0860538

a. The SVS w.r.t. TS or RB shall describe which are the tasks and the items under tests, as well as criteria to be utilized.

<5.2> Features to be tested

ECSS-E-ST-40_0860539

a. The SVS w.r.t. TS or RB shall describe all the features to be tested, making references to the applicable documentation.

<5.3> Features not to be tested

ECSS-E-ST-40_0860540

a. The SVS w.r.t. TS or RB shall describe all the features and significant combinations not to be tested.

<5.4> Test pass - fail criteria

ECSS-E-ST-40_0860541

a. The SVS w.r.t. TS or RB shall describe the general criteria to be used to determine whether or not tests are passed.

<5.5> Items that cannot be validated by test

ECSS-E-ST-40_0860542

a. The SVS w.r.t. TS or RB shall list the tasks and items under tests that cannot be validated by test.

ECSS-E-ST-40_0860543

b. Each of them shall be properly justified

ECSS-E-ST-40_0860544

c. For each of them, an analysis, inspection or review of design shall be proposed.

<5.6> Manually and automatically generated code

ECSS-E-ST-40_0860545

a. The SVS shall address separately the activities to be performed for manually and automatically generated code, although they have the same objective (ECSS-Q-ST-80 clause 6.2.8.2 and 6.2.8.7).

<6> Software validation testing specification design

<6.1> General

ECSS-E-ST-40_0860546

a. The SVS w.r.t. TS or RB shall provide the definition of software validation testing specification design, giving the design grouping criteria such as function, component or equipment management.

ECSS-E-ST-40_0860547

b. For each identified test design, the SVS w.r.t. TS or RB shall provide the information given in <6.2>.

<6.2> Organization of each identified test design

NOTE The SVS w.r.t. TS or RB provides the definition of each validation test design, detailed as follows

<6.2.1> General

ECSS-E-ST-40_0860548

a. The SVS w.r.t. TS or RB shall briefly describe the test design.

<6.2.2> Features to be tested

ECSS-E-ST-40_0860549

a. The SVS w.r.t. TS or RB shall describe the test items and the features to be tested.

ECSS-E-ST-40_0860550

b. Reference to appropriate documentation shall be performed and traceability information shall be provided.

<6.2.3> Approach refinements

ECSS-E-ST-40_0860551

a. The SVS w.r.t. TS or RB shall describe the test approach implemented for the specific test design and the specific test class implemented.

ECSS-E-ST-40_0860552

b. The description specified in a. shall provide the rationale for the test case selection and grouping into test procedures.

ECSS-E-ST-40_0860553

c. The method for analysing test results shall be identified (e.g. compare with expected output, and compare with old results).

ECSS-E-ST-40_0860554

d. Configuration of the facility (both hardware and software) to be used to execute the identified test shall be described.

<7> Software validation test case specification

<7.1> General

ECSS-E-ST-40_0860555

a. The SVS w.r.t. TS or RB shall provide the identification of software validation test cases.

ECSS-E-ST-40_0860556

b. For each identified test case, the SVS w.r.t. TS or RB shall provide the information given in 7.2

<7.2> Organization of each identified test case

NOTE The SVS w.r.t. TS or RB provides the definition of each validation test case, detailed as follows.

<7.2.1> Test case identifier

ECSS-E-ST-40_0860557

a. The SVS w.r.t. TS or RB shall describe the test case uniquely.

ECSS-E-ST-40_0860558

b. A short description of the test case purpose shall be provided.

<7.2.2> Inputs specification

ECSS-E-ST-40_0860559

a. The SVS w.r.t. TS or RB shall describe, for each test case, the inputs to execute the test case.

<7.2.3> Outputs specification

ECSS-E-ST-40_0860560

a. The SVS w.r.t. TS or RB shall describe, for each test case, the expected outputs.

<7.2.4> Test pass - fail criteria

ECSS-E-ST-40_0860561

a. The SVS w.r.t. TS or RB shall describe, for each test case, the criteria to decide whether the test has passed or failed.

<7.2.5> Environmental needs

ECSS-E-ST-40_0860562

a. The SVS w.r.t. TS or RB shall describe:

   1. the exact configuration and the set up of the facility used to execute the test case as well as the utilization of any special test equipment (e.g. bus analyser);

2. the configuration of the software utilized to support the test conduction (e.g. identification of the simulation configuration);

<7.2.6> Special procedural constraints(ECSS-Q-ST-80 clause 6.3.5.25)

ECSS-E-ST-40_0860563

a. The SVS w.r.t. TS or RB shall describe any special constraints on the used test procedures.

<7.2.7> Interfaces dependencies

ECSS-E-ST-40_0860564

a. The SVS w.r.t. TS or RB shall list all the test cases to be executed before this test case.

<8> Software validation test procedures

<8.1> General

ECSS-E-ST-40_0860715

a. This part of the DRD may be placed in a different document, if agreed with the customer.

NOTE Procedures are not always attached to each test case

ECSS-E-ST-40_0860566

b. The SVS w.r.t. TS or RB shall provide the identification of software validation test procedures.

ECSS-E-ST-40_0860567

c. For each identified validation test procedure, the SVS w.r.t. TS or RB shall provide the information presented in 8.2

<8.2> Organization of each identified test procedure

NOTE The SVS w.r.t. TS or RB provides the description of each identified validation test procedure, detailed as follows.

<8.2.1> Test procedure identifier

ECSS-E-ST-40_0860568

a. The SVS w.r.t. TS or RB shall identify each test procedure uniquely.

<8.2.2> Purpose

ECSS-E-ST-40_0860569

a. The SVS w.r.t. TS or RB shall describe the purpose of each test procedure.

ECSS-E-ST-40_0860570

b. A reference to each test case used by the test procedure shall be given.

<8.2.3> Procedure steps

ECSS-E-ST-40_0860571

a. The SVS w.r.t. TS or RB shall describe every step of each procedure execution:

1. log: describe any special methods or format for logging the results of test execution, the incidents observed, and any other event pertinent to this test;
2. set up: describe the sequence of actions necessary to set up the procedure execution;
3. start: describe the actions necessary to begin the procedure execution;
4. proceed: describe the actions necessary during the procedure execution;
5. test result acquisition: describe how the test measurements is made;
6. shut down: describe the action necessary to suspend testing when interruption is forced by unscheduled events;
7. restart: identify any procedural restart points and describe the actions necessary to restart the procedure at each of these points;
8. wrap up: describe the actions necessary to terminate testing.

<8.2.4> Test script

ECSS-E-ST-40_0860572

a. The SVS w.r.t. TS or RB shall list all the test script used to execute the test case.

NOTE The test scripts can be collected in an appendix.

<9> Software validation analysis, inspection, review of design

ECSS-E-ST-40_0860573

a. The SVS w.r.t. TS or RB shall include, for each items where it can be justified that a test is not possible, another validation method based on analysis, inspection, review of design.

<10> Validation test platform requirements

ECSS-E-ST-40_0860574

a. The SVS w.r.t. TS or RB shall list the validation requirements related to the validation test platform to be used (for example, benches or simulators capabilities and their representativeness with respect to e.g. real time constraints, target or real hardware equipment on which the software is specified to operate).

<11> Software validation specification additional information

ECSS-E-ST-40_0860575

a. The following additional information shall be included in the SVS w.r.t. TS or RB:

1. Test/analysis/inspection/review of design to requirement traceability matrix,
2. Requirement to test/analysis/inspection/review of design traceability matrix,
3. Test procedures to test cases traceability matrix,
4. Test cases to test procedures traceability matrix,
5. Test scripts,
6. Detailed test procedures.

NOTE 1 This information can be given in separate appendices.

NOTE 2 One test design uses one or more test cases.

NOTE 3 One test procedure execute one or more test cases.

NOTE 4 Traceability matrixes include the title of the requirement or test in addition to its number for readability purpose.

L.2.2 Special remarks

None.

Annex M (normative) Software verification report (SVR) - DRD

M.1 DRD identification

M.1.1 Requirement identification and source document

The software verification report (SVR) is called from the normative provisions summarized in Table M-1.

Table M-1: SVR traceability to ECSS-E-ST-40 and ECSS-Q-ST-80 clauses

ECSS Standard	Clauses	DRD section
ECSS-E-ST-40	5.3.8.1a	<7>
	5.3.8.2a	<5.2>
	5.3.8.2b	<5.2>
	5.7.3.5	<4.5>a.1.
	5.8.3.1	<4.2>
	5.8.3.2a eo a	<4.3.1>
	5.8.3.2a eo b	<4.3.2>
	5.8.3.3a eo a	<4.3.1>
	5.8.3.3a eo b	<4.3.2>
	5.8.3.4a eo a.	<4.4>a.1.
	5.8.3.4a eo b	<4.4>a.2.
	5.8.3.5a eo a	<4.4>a.1.
	5.8.3.5a eo b	<4.4>a.2.
	5.8.3.5	<4.4>a.2., <4.5>a.2.
	5.8.3.5c	<4.4>a.2., <4.5>a.2.
	5.8.3.5d	<4.4>a.2., <4.5>a.2.
	5.8.3.5e	<4.4>a.2., <4.5>a.2.
	5.8.3.5f	<4.4>a.2.
	5.8.3.6a eo a	<4.4>a.1.
	5.8.3.6a eo b	<4.4>a.2.
	5.8.3.7	<4.4>a.2.
	5.8.3.8a eo a	<4.6>a.1.

	5.8.3.8a eo b	<4.6>a.1.
	5.8.3.8b eo a	<4.6>a.2.
	5.8.3.8b eo b	<4.6>a.2.
	5.8.3.10	<4.3.2>, <4.4>a.2., <4.5>a.2., <4.6>a.2.
	5.8.3.11a	<5>
	5.8.3.11b	<5>
	5.8.3.11c	<5>
	5.8.3.12a	<5>
	5.8.3.12b	<5>
	5.8.3.12c	<5>
	5.8.3.13a	<4.3.2>
	5.8.3.13b	<4.3.2>
	5.8.3.13	<4.4>a.2.
ECSS-Q-ST-80	6.2.6.5	<4.4>a.2.
	6.2.6.6	<4.4>a.2.
	7.1.7	<6>
	7.2.3.6	<4.6>a.2.
eo = Expected Output		

M.1.2 Purpose and objective

The software verification report is a constituent of the design justification file (DJF). Its purpose is to present gathered results of all the software verification activities that have to be executed along the software development life cycle according to the SVerP. It is organized per process, except for the timing and sizing issues which are gathered in a separate section. Each process verification report can be placed into a separate document.

M.2 Expected response

M.2.1 Scope and content

<1> Introduction

ECSS-E-ST-40_0860576

a. The SVR shall contain a description of the purpose, objective, content and the reason prompting its preparation.

<2> Applicable and reference documents

ECSS-E-ST-40_0860577

a. The SVR shall list the applicable and reference documents to support the generation of the document.

<3> Terms, definitions and abbreviated terms

ECSS-E-ST-40_0860578

a. The SVR shall include any additional terms, definition or abbreviated terms used not already defined in any applicable or reference documentation.

<4> Verification activities reporting and monitoring

<4.1> General

ECSS-E-ST-40_0860579

a. The SVR shall address separately the activities to be performed for manually and automatically generated code.

<4.2> Software related system requirements process verification (for the SRR)

ECSS-E-ST-40_0860580

a. The SVR shall include the report of the verification of the requirements baseline and the interface requirements specification as specified in 5.8.3.1.

ECSS-E-ST-40_0860581

b. If system models are available, a model checking report (e.g. data, event, failure) shall be included in the SVR.

<4.3> Software requirements and architecture engineering process verification (for the PDR)

<4.3.1> Traceability (when not already included in related software requirements, interface and design documents)

ECSS-E-ST-40_0860582

a. The SVR shall present the following traceability matrices:

- software requirements to system requirements,
- software architectural design to requirements.

<4.3.2> Feasibility

ECSS-E-ST-40_0860583

a. The SVR shall present in gathering all the specific verification reports that have been planned to be provided w.r.t. the SVerP, including:

- Software requirements verification as per 5.8.3.2;

- — HMI evaluation by e.g. mock-up as per ECSS-E-ST-10-11;
- — Behavioural verification of the logical model and architectural design verification as per 5.8.3.13a. and b;
- — Verification of the software architectural and interface design as per 5.8.3.3;
- — Architectural design behavioural model checking;
- — Verification of software documentation as per 5.8.3.10;
- — Other specific inspections, analyses or review of design report (e.g. numerical accuracy, technical risks analysis, evaluation of reuse potential);
- — Others specific verification related to RAMS requirements (e.g. analysis reports using HSIA, SFTA, SFMECA).

<4.4> Software design and implementation engineering process verification (for CDR)

ECSS-E-ST-40_0860584

a. The SVR shall present in gathering all the specific verification reports that have been planned to be provided w.r.t. the SVerP, including:

1. Traceability (when not already included in related software design documents or software code), presenting the following traceability matrices:
   - o software detailed design to software architectural design,
   - o Software code to software detailed design,
   - o Software unit test to requirements, design.
2. Feasibility, presenting in gathering all the specific verification reports that have been planned to be provided w.r.t. the SVerP, including e.g.:
   - o Software detailed design verification as per 5.8.3.4;
   - o Design model checking (including behavioural verification as per 5.8.3.13b.);
   - o Software code verification as per 5.8.3.5a;
   - o Structural code coverage achievement;
   - o Deactivated code verification as per ECSS-Q-ST-80 6.2.6.5;
   - o Configurable code verification as per ECSS-Q-ST-80 6.2.6.6;
   - o Source code robustness verification;
   - o Verification of software unit testing as per 5.8.3.6;
   - o Verification of software integration as per 5.8.3.7;
   - o Verification of software documentation as per 5.8.3.10;
   - o Others specific inspections, analyses or review of design report (e.g. technical risks analysis, evaluation of reuse potential);
   - o Others specific verification related to RAMS requirements (e.g. analysis reports using HSIA, SFTA, SFMECA).

<4.5> Software delivery and acceptance process verification (for QR and AR)

ECSS-E-ST-40_0860585

a. The SVR shall present in gathering all the specific verification reports that have been planned to be provided w.r.t. the SVerP, including:

1. Traceability (when not already included in related software acceptance documents), presenting the following traceability matrices:
   - Software acceptance testing to requirements baseline.
2. Feasibility, presenting in gathering all the specific verification reports that have been planned to be provided w.r.t. the SVerP, including:
   - Structural code coverage achievement (update for QR and AR);
   - Verification of software documentation as per 5.8.3.10;
   - Others specific verification related to RAMS design (e.g. unit and integration testing coverage ratio).

<4.6> Software validation process verification (for CDR, QR, AR)

ECSS-E-ST-40_0860586

a. The SVR shall present in gathering all the specific verification reports that have been planned to be provided w.r.t. the SVerP, including:

1. Traceability (when not already included in related software validation specification), presenting the following traceability matrices:
   - software validation specifications to TS,
   - software validation specifications to RB.
2. Feasibility, presenting in gathering all the specific verification reports that have been planned to be provided w.r.t. the SVerP, including e.g.:
   - Verification of software validation w.r.t. TS as per 5.8.3.8a.,
   - Verification of software validation w.r.t. RB as per 5.8.3.8b.,
   - Verification of software documentation as per 5.8.3.10,
   - Verification of testing as per ECSS-Q-ST-80 clause 7.2.3.6.

<4.7> Software quality requirements verification

ECSS-E-ST-40_0860587

a. The SVR shall present in gathering all the specific verification reports related to software quality that have been planned to be provided in the SVerP. This include in particular the verification of the software quality requirements according to ECSS-Q-ST-80 clause 6.2.6.1, and the verification of the application of the chosen measures to handle automatically generated code.

NOTE Deactivated and configurable code verification reports are in section <4.4>a.2. Numerical accuracy report is in section <6> of this DRD.

<5> Margin and technical budget status

NOTE This section is often placed in a separate document named STSB (Software Timing and Sizing Budget).

<5.1> Technical budgets and margins computation

ECSS-E-ST-40_0860588

a. The SVR shall include the way to compute the technical budgets and margins.

<5.2> Software budget (sizing and timing)

ECSS-E-ST-40_0860589

a. The status of margins regarding the technical budgets shall be presented in the SVR at each milestone, describing the utilized analytical hypothesis.

ECSS-E-ST-40_0860590

b. The margins shall be established by estimation for PDR, by analysis of design after detailed design, and consolidated by performance measurements commensurate with the software implementation for CDR, QR and AR.

ECSS-E-ST-40_0860591

c. The SVR shall include at PDR:

1. the memory size for static code size, static data size and stack size;
2. the processing unit utilization;
3. the deadline fulfilment, the margin available for every deadline in the worst case and, if feasible, the jitter in the nominal case.

ECSS-E-ST-40_0860592

d. The SVR shall include after detailed design:

1. the memory size refined for static code size, static data size and stack size expressed on a thread basis, measuring them per lowest level design component;
2. the processing unit utilization, refined, considering the worst case execution time of each lowest level design component having its own control flow (therefore including the call to the protected objects) (expressed in time and in percentage of a reference period);
3. the deadline fulfilment.

NOTE The worst case execution time of each lowest level design component having its own control flow is multiplied by the number of times this component is executed per second. The resulting quantity is summed over all other design components. The result is the estimated percentage processor utilization.

<5.3> Schedulability simulation and analyses

ECSS-E-ST-40_0860593

a. The SVR shall include the result of the schedulability analysis or the schedulability simulation, based on:

1. estimated values at PDR,
2. refined values after detailed design,
3. measured values at CDR.

NOTE An example of schedulability analysis report is a table with the following columns:

- process name,
- P: process priority,
- C: process worst case execution time,
- T: process period,
- D: process deadline,
- I: process interference time, the time that the process can be interrupted by processes of higher priority,
- B: process blocking time, the time that the process can be blocked on a protected object access by a process of lower priority,
- S: process schedulability factor in percentage, computed as the sum of C, I and B, this sum divided by D.

<6> Numerical accuracy analysis

ECSS-E-ST-40_0860594

a. The SVR shall include the estimation and the verification of the numerical accuracy.

<7> Interface timing analysis

ECSS-E-ST-40_0860798

a. The SVR shall include the estimation and the verification of the throughput and latency of the external interfaces controlled by the software as per 5.3.8.1.

M.2.2 Special remarks

None.

Annex N (normative) Software reuse file (SRF) - DRD

N.1 DRD identification

N.1.1 Requirement identification and source document

The software reuse file (SRF) is called from the normative provisions summarized in Table N-1.

Table N-1: SRF traceability to ECSS-E-ST-40 and ECSS-Q-ST-80 clauses

ECSS Standard	Clauses	DRD section
ECSS-E-ST-40	5.4.3.6a	<4>, <6>b.3.
	5.4.3.6b	<4>, <5>, <6>
	5.4.3.7	<4>, <5>
ECSS-Q-ST-80	5.4.1.2a	All
	6.2.7.2a eo b.	<6>
	6.2.7.3a eo b	<4>, <5>
	6.2.7.4a eo b	<5>b., <5>c.
	6.2.7.5a eo b	<6>b.2., <6>b.3., <7>
	6.2.7.6a	<4>, <5>
	6.2.7.7a	<8>
	6.2.7.8b	<8>
	6.2.7.9a	All
	6.2.7.11a	<9>
eo = Expected Output		

N.1.2 Purpose and objective

The software reuse file is a constituent of the design justification file (DJF). Its purpose is to document the analysis to be performed on existing software intended to be reused.

The global objectives of the software reuse file are to document all the information used to decide about the reuse (or not) of existing software and to plan the specific actions undertaken to ensure that the reused software meets the project requirements.

The SRF is also used to document software developed for intended reuse, such that it is ready when the software is actually reused.

N.2 Expected response

N.2.1 Scope and content

<1> Introduction

ECSS-E-ST-40_0860595

a. The SRF shall contain a description of the purpose, objective, content and the reason prompting its preparation.

<2> Applicable and reference documents

ECSS-E-ST-40_0860596

a. The SRF shall list the applicable and reference documents to support the generation of the document.

<3> Terms, definitions and abbreviated terms

ECSS-E-ST-40_0860597

a. The SRF shall include any additional terms, definition or abbreviated terms used not already defined in any applicable or reference documentation.

<4> Presentation of the software intended to be reused

ECSS-E-ST-40_0860598

a. The SRF shall describe the technical and management information available on the software intended for reuse.

ECSS-E-ST-40_0860599

b. For each software item, the SRF shall provide (or state the absence of) the following information:

1. software item name and main features;
2. software item developer identification;
3. considered version and list of components;
4. licensing conditions;
5. industrial property and exportability constraints, if any;
6. implementation language;
7. development and execution environment (e.g. platform, and operating system);
8. applicable dispositions for warranty, maintenance, installation and training;
9. commercial software necessary for software execution, if any;
10. size of the software (e.g. number of source code lines, and size of the executable code).

<5> Compatibility of existing software with project requirements

ECSS-E-ST-40_0860600

a. The SRF shall describe which part of the project requirements (RB) are intended to be implemented through software reuse.

ECSS-E-ST-40_0860601

b. For each software item, the SRF shall provide the availability and quality status (completeness, correctness, etc.) of the following information:

1. software requirements documentation;
2. software architectural and detailed design documentation;
3. forward and backward traceability between system requirements;
4. software requirements, design and code;
5. unit tests documentation and coverage;
6. integration tests documentation and coverage;
7. validation documentation and coverage;
8. verification reports;
9. performance (e.g. memory occupation, processing unit load);
10. operational performances;
11. residual non conformance and waivers;
12. user operational documentation (e.g. user manual);
13. code quality (adherence to coding standards, metrics).

ECSS-E-ST-40_0860602

c. For each of the points in <5>b, the SRF shall document the quality level of the existing software with respect to the applicable project requirements, according to the criticality of the system function implemented.

<6> Software reuse analysis conclusion

ECSS-E-ST-40_0860603

a. The SRF shall document the results of the software reuse analysis.

ECSS-E-ST-40_0860604

b. For each software item, the SRF shall provide the following information:

1. decision to reuse or not reuse, based on the information provided in previous chapters;
2. estimated level of reuse;
3. assumptions and methods applied when estimating the level of reuse.

<7> Detailed results of evaluation

ECSS-E-ST-40_0860605

a. The SRF shall include the detailed results of the evaluation.

NOTE The detailed results of the evaluation can be presented in an appendix.

<8> Corrective actions

ECSS-E-ST-40_0860606

a. The SRF shall document any corrective actions identified to ensure that the software intended for reuse meets the applicable project requirements.

ECSS-E-ST-40_0860607

b. The SRF shall document the detailed results of the implementation of the identified corrective actions.

<9> Configuration status

ECSS-E-ST-40_0860608

a. The SRF shall include the detailed configuration status of the reused software baseline.

N.2.2 Special remarks

None.

Annex O (normative) Software development plan (SDP) - DRD

O.1 DRD identification

O.1.1 Requirement identification and source document

The software development plan (SDP) is called from the normative provisions summarized in Table O-1.

Table O-1: SDP traceability to ECSS-E-ST-40 and ECSS-Q-ST-80 clauses

ECSS Standard	Clauses	DRD section
ECSS-E-ST-40	5.3.2.1a	<5.2.1>
	5.3.2.1b	<5.2.1>
	5.3.2.1c	<5.1>, <5.3>, <5.4>
	5.3.2.1d	<5.2.3>, <5.5>
	5.3.2.2a	<5.2.1>
	5.3.2.3a	<4.8>
	5.3.2.4a	<5.2>
	5.3.2.4b	<5.3>
	5.3.2.4c	<5.3>
	5.3.2.4d	<5.4>
	5.3.3.2a	<5.2.3>
	5.3.3.3a	<5.2.3>
	5.3.3.3b	<5.2.3>
	5.3.3.3c	<5.2.3>
	5.3.5.3a	<5.2.3>
	5.3.5.4a	<5.2.3>
	5.3.6.1a.	<5.2.2>
	5.3.6.1b	<5.2.2>
	5.3.6.2a	<5.2.2>
	5.3.9.1a	<5.6>
	5.3.9.2a	<5.6>
ECSS-Q-ST-80	5.5.2	<4.8>
	5.6.2.1	<5.4>
	5.6.2.2	<5.4>
	6.3.4.5	<5.4>a.

O.1.2 Purpose and objective

The software development plan is a constituent of the management file (MGT). Its purpose is to describe the established management and development approach for the software items to be defined by a software supplier to set up a software project in accordance with the customer requirements.

O.2 Expected response

O.2.1 Scope and content

<1> Introduction

ECSS-E-ST-40_0860609

a. The SDP shall contain a description of the purpose, objective, content and the reason prompting its preparation.

<2> Applicable and reference documents

ECSS-E-ST-40_0860610

a. The SDP shall list the applicable and reference documents to support the generation of the document.

<3> Terms, definitions and abbreviated terms

ECSS-E-ST-40_0860611

a. The SDP shall include any additional terms, definition or abbreviated terms used not already defined in any applicable or reference documentation.

<4> Software project management approach

<4.1> Management objectives and priorities

ECSS-E-ST-40_0860612

a. The SDP shall describe the management objectives of the software project and associated priorities.

<4.2> Master schedule

ECSS-E-ST-40_0860613

a. The SDP shall make a reference to the general project master schedule.

<4.3> Assumptions, dependencies and constraints

ECSS-E-ST-40_0860614

a. The SDP shall state:

1. the assumptions on which the plan is based;
2. the external events the project is dependent upon;
3. constraints on the project;
4. technical issues.

NOTE Technical issues are only mentioned if they have an effect on the plan. Assumptions, dependencies and constraints are often difficult to distinguish. The best approach is not to categorize them but to list them. For example:

- limitations on the budget;
- schedule constraints (e.g. launch dates, delivery dates);
- constraints on the location of staff (e.g. the obligation to work at developer’s premises);
- commercial hardware or software used by the system;
- constraints on the development of hardware functions in the case of a hardware/software co-engineering development (e.g. interfaces, simulations, prototypes);
- availability of simulators and others test devices;
- availability of external systems with which the system interfaces.

<4.4> Work breakdown structure

ECSS-E-ST-40_0860615

a. The SDP shall list the activities to be performed in order to develop the software configuration item, and include or reference the work package description.

NOTE 1 See ECSS-M-ST-10 for further explanation.

NOTE 2 Sometimes the adequate elementary tasks can be identified only if several levels of activities breakdown are performed.

<4.5> Risk management

ECSS-E-ST-40_0860616

a. The SDP shall describe the contribution of the software engineering function to the project risk management approach.

NOTE See ECSS-M-ST-80 for further explanations.

<4.6> Monitoring and controlling mechanisms

ECSS-E-ST-40_0860617

a. The SDP shall describe the monitoring mechanisms for managing the work.

NOTE 1 The SDP apply to both the customer relationships and the supplier's relationships.

NOTE 2 Monitoring mechanisms include progress report, progress meeting, action item lists.

<4.7> Staffing plan

ECSS-E-ST-40_0860618

a. The SDP shall describe the roles and skills of staff involved in the project, the organisational structure, boundaries and interface, the external interface responsibles, and the resources.

<4.8> Software procurement process

ECSS-E-ST-40_0860619

a. The SDP shall describe the software procurement process implementation, and include here or in annex the procured software component list.

<4.9> Supplier management

ECSS-E-ST-40_0860620

a. The SDP shall describe the supplier management approach.

NOTE The software management aspects specified in <4.1> to <4.8> can be fully described in the SDP or, at higher level, in a project management plan according to the ECSS-M Standards

<5> Software development approach

<5.1> Strategy to the software development

ECSS-E-ST-40_0860621

a. The SDP shall describe the overall strategy to the software development.

NOTE 1 An activity diagram can be included.

NOTE 2 This includes the description of the hardware/software co-engineering strategy when software is tightly linked to hardware functions.

<5.2> Software project development life cycle

<5.2.1> Software development life cycle identification

ECSS-E-ST-40_0860622

a. The SDP shall describe the software development life cycle.

ECSS-E-ST-40_0860623

b. Definition of the selected life cycle paradigm as well as the adopted software versioning approach shall be included.

NOTE Typical life cycles are waterfall, incremental, evolutionary, agile-based.

ECSS-E-ST-40_0860624

c. The SDP shall cover the implementation of all the activities and tasks relevant to the involved software processes, including:

- system engineering processes related to software;
- software requirement & architecture engineering process;
- software design and implementation engineering process;
- software validation process;
- software verification process;
- software delivery and acceptance;
- software operation process;
- software maintenance process and its interface with development (documents to be handed over, tools to be maintained);
- software management process.

<5.2.2> Relationship with the system development cycle

ECSS-E-ST-40_0860625

a. The SDP shall describe the phasing of the software life cycle to the system development life cycle.

NOTE A process model representation can be used.

<5.2.3> Reviews and milestones identification and associated documentation

ECSS-E-ST-40_0860626

a. The SDP shall describe scope and purpose of each identified review, relevant deliverable and expected outputs.

ECSS-E-ST-40_0860627

b. For technical reviews, the SDP shall specify the applicable level of formalism.

ECSS-E-ST-40_0860628

c. The role of involved parties or organizations at each review shall be described.

<5.3> Software engineering standards and techniques

ECSS-E-ST-40_0860629

a. The SDP shall describe (or provide references to their description of) the applied methodologies and list the standards for each software process and relevant activity.

ECSS-E-ST-40_0860630

b. The requirements analysis method used shall be listed and referenced.

NOTE Reference to applied literature or other standards can be described here.

ECSS-E-ST-40_0860631

c. Any adaptation of the requirements analysis method shall be described or referenced.

ECSS-E-ST-40_0860632

d. The selected design (architectural design and detailed design) methods shall be stated and referenced.

NOTE Reference to applied literature or other standards can be described here.

ECSS-E-ST-40_0860633

e. The parts of the software subject to auto-code generation shall be identified at SRR and confirmed with the definition of interface and resource allocation at PDR.

ECSS-E-ST-40_0860634

f. The specificity of automatic code generation tool chains shall be considered, in particular for maintenance.

ECSS-E-ST-40_0860635

g. Any adaptation of the design method (e.g. deviations, extensions, and avoidance of utilization of some methodology features) shall be described or referenced.

ECSS-E-ST-40_0860636

h. Any HMI standard to be applied to the software development, if code generators are utilized (e.g. constraints to be imposed to use of generators in terms of allowed specific features) shall be documented.

ECSS-E-ST-40_0860637

i. The selected software delivery format shall be described or referenced.

<5.4> Software development and software testing environment

ECSS-E-ST-40_0860638

a. This SDP shall describe the software development environment and testing environment, including the evidence of its suitability and the programming language selection suitability.

ECSS-E-ST-40_0860639

b. Hardware platforms and selected software tools to be utilized for the software development and testing shall be described and include or reference the justification of their selection with respect to the relevant software methodology and standards.

NOTE This covers, in particular, the tools used to configure the software for a particular mission with the parameters of the mission database.

ECSS-E-ST-40_0860640

c. The information in <5.4>b. shall, as a minimum, include:

1. the compiler and cross compiler system;
2. the requirements analysis tool;
3. the tools utilized in the software image generation;
4. the configuration management tools;
5. the software design tools;
6. the software static analysis tools;
7. the software test scripts language tools;
8. the software testing tools (debuggers, in circuit emulator, bus analyser).

<5.5> Software documentation plan

<5.5.1> General

ECSS-E-ST-40_0860641

a. The SDP shall describe or refer to all documentation relevant to the project and the documentation standards applied to the software project.

<5.5.2> Software documentation identification

ECSS-E-ST-40_0860642

a. The SDP, for each document to be produced (both internal documents and deliverable), shall include the documentation plan stating:

1. the documentation file;
2. the document name;
3. the delivery requirements;
4. the review requirements;
5. the approval requirements.

<5.5.3> Deliverable items

ECSS-E-ST-40_0860643

a. The SDP shall list the items to be delivered.

ECSS-E-ST-40_0860644

b. The SDP shall clearly address deliverable items internal to the software development organization (what, when and how).

ECSS-E-ST-40_0860799

c. The SDP shall identify the information to be provided for each software delivery including at least:

   1. Sender and receiver of the delivery: from supplier to customer, from customer to final user, or both;

   2. Perimeter of the delivery, including but not limited to:

      (a) Purpose of the delivery, i.e. what the software version will be used for.

      (b) Functionality included.

      (c) Level of maturity in terms of testing/qualification.

      (d) Limitations of the delivery listing shortcomings and known issues.

   3. Composition and level of formality of the corresponding SW DRB.

      NOTE This information is also expected for deliveries defined during the life cycle, i.e. not initially planned.

<5.5.4> Software documentation standards

ECSS-E-ST-40_0860645

a. The SDP shall describe the documentation standards applicable to the project.

ECSS-E-ST-40_0860646

b. Any tailoring to applicable documentation standards shall be detailed in this clause.

<5.6> This Standard’s tailoring traceability

ECSS-E-ST-40_0860647

a. The SDP shall include the coverage matrix of the applicable tailoring of ECSS-E-ST-40 clause 5, or provide a reference to it.

O.2.2 Special remarks

None.

Annex P (normative) Software review plan (SRevP) - DRD

P.1 DRD identification

P.1.1 Requirement identification and source document

The software review plan (SRevP) is called from the normative provisions summarized in Table P-1.

Table P-1: SRevP traceability to ECSS-E-ST-40 and ECSS-Q-ST-80 clauses

ECSS Standard	Clauses	DRD section
ECSS-E-ST-40	5.3.3.2b	All

P.1.2 Purpose and objective

The software review plan is a constituent of the design justification file (DJF). Its purpose is defined in ECSS-M-ST-10-01. This DRD is the tailoring of the ECSS-M-ST-10-01 standard for software.

P.2 Expected response

P.2.1 Scope and content

<1> Introduction

ECSS-E-ST-40_0860648

a. The SRevP shall contain a description of the purpose, objective, content and the reason prompting its preparation.

<2> Applicable and reference documents

ECSS-E-ST-40_0860649

a. The SRevP shall list the applicable and reference documents to support the generation of the document.

<3> Terms, definitions and abbreviated terms

ECSS-E-ST-40_0860650

a. The SRevP shall include any additional terms, definition or abbreviated terms used not already defined in any applicable or reference documentation.

<4> Review title and project

<4.1> Exact name

ECSS-E-ST-40_0860651

a. This section shall define the exact name of the review

<4.2> System or product subject to review

ECSS-E-ST-40_0860652

a. This section shall describe the product subject to review, and its current development stage or the one expected for this review.

<5> Reference documents

ECSS-E-ST-40_0860653

a. The SRevP shall list all project documentation applicable to the review. Note: the documentation subject for review is detailed in section 10 of this SRevP.

ECSS-E-ST-40_0860654

b. Review data package documentation shall be according to the deliveries for every review in the Software life cycle chosen (see Table A-1: ECSS-E-ST-40 and ECSS-Q-ST-80 Document requirements list (DRL)).

<6> Review objectives

ECSS-E-ST-40_0860655

a. This section shall describe the purpose of the review.

NOTE 1 Typical objectives of the software system requirements (SRR) review are:

- Agree with the customer or their representatives that all requirements captured in the requirements baseline are commonly understood and agreed.
- Release the Requirements Baseline.
- Check the suitability of the draft software development plan including the software planning elements.

- Verify the consistency of the software planning elements with respect to the upper level planning.
- Check the status of the ISVV activities.
- Ensure that software product assurance activities are performed.
- Evaluate the readiness to proceed to the next phase.

NOTE 2 Typical objectives of the software requirement review (SWRR) (held as anticipation of the PDR) are:

- Agree with the customer or their representatives that all requirements with respect to the requirements baseline are captured in the technical specification.
- Release the Technical Specification.
- Verify the technical budget and margins estimations.
- Evaluate known unsolved issues which can have major impacts.
- Verify the quality assurance reports.
- Evaluate the readiness to proceed to the next phase.

NOTE 3 Typical objectives of the software preliminary design review (PDR) are:

- Agree with the customer or their representatives that all requirements with respect to the requirements baseline are captured in the technical specification.
- Release the Technical Specification.
- Verify the software development approach and release the relevant plan.
- Establish the software product assurance approach and release the relevant plan.
- Establish the software configuration management approach and release the relevant plan.
- Establish the software verification and validation approach and release the relevant plan.
- Release the software architecture.
- Verify the technical budget and margins estimations.
- Check the integration strategy.

- Evaluate the potential re-use of the software if applicable.
- Evaluate known unsolved issues that can have major impacts.
- Check the status of the ISVV activities.
- Verify the quality assurance reports.
- Evaluate the security verification and validation process implementation.
- Evaluate the readiness to proceed to the next phase.

NOTE 4 Typical objectives of the software detailed design review (DDR) (held as anticipation of the CDR) are:

- Release the detailed design.
- Verify the software technical budget status (e.g. PU and memory).
- Baseline of the detailed design (i.e. baseline the software detailed design).
- Check the adequacy of the software units and integration plans.
- Release the Software Reuse File.
- Evaluate the potential re-use of the software.
- Verify the feasibility of integration and testing.
- Evaluate known unsolved issues which can have major impact.
- Verify the quality assurance reports.
- Evaluate the readiness to proceed to the next phase.

NOTE 5 Typical objectives of the software validation specification review (SVSR) are:

- Verify that the Software Validation Specification is complete w.r.t. the technical specification or requirements baseline.

NOTE 6 Typical objectives of the software test readiness review (TRR) are:

- Baseline the testing, analysis, inspection or review of design (e.g. Software Validation Specification w.r.t. the technical specification or requirements baseline).
- Release the design documents.
- Verify the integration and TS/RB-validation facilities.
- Release the Unit and Integration Test Results.

- Check the testing facilities configuration.
- Verify that software documentation, software code, procured software and support software and facilities are under proper configuration control.
- Baseline the testing configuration.
- Verify the quality assurance reports.
- Check the status of all SPRs and NCRs.
- Evaluate the readiness to proceed to testing.

NOTE 7 Typical objectives of the review of the test review board (TRB) are:

- Verify the test results with respect to the testing specification and plans.

NOTE 8 Typical objectives of the software delivery review board (SW DRB) are:

- Verify that the destination environment is ready to receive the software.
- Verify that the delivery is complete, in line with the agreed scope.
- Verify that the delivered software and accompanying documentation are under configuration control.
- Verify the software delivery media and delivery method.
- Verify that the software installation procedure and required support for the installation are available.
- Assessment of the validation status of the software with respect to the purpose of the delivery.
- Verify the quality of the delivery with respect to the agreed scope.
- Evaluate known unresolved issues and their impact on the delivery.

NOTE 9 Typical objectives of the software critical design review (CDR) are:

- Baseline the detailed design (including the verification reports and technical budget report).
- Check the adequacy of the software units and integration plans and of the included unit and integration test procedures.
- Review and baseline the SValP approach and relevant plan.

- Verify the Software Reuse File, evaluate the potential re-use of the software intended for reuse.
- Check the validation specification w.r.t. the technical specification.
- Assess the unit and integration test results, including as-run procedures.
- Ensure that all the Technical Specification has been successfully validated (validation report) and verified (including technical budget, memory and PU, and code coverage).
- Verify that the Software Configuration Item under review is a formal version under configuration control.
- Release the software user manual.
- Evaluate known unresolved issues that can have major impact and resolution plan identification.
- Check the status of the ISVV activities.
- Verify the quality assurance reports.
- Check the RB-validation facility.
- Baseline the Validation Specification against the RB.
- Evaluate the security integration and validation activities, including security testing.
- Evaluate the readiness to proceed to the next phase.

NOTE 10Typical objectives of the software qualification review (QR) are:

- Verify that the software meets all of its specified requirements, and in particular, that verification and validation processes have been completed successfully.
- Check the RB-validation test, analysis, inspection or review of design results, including as-run procedures.
- Verify that all the Requirements Baseline and interfaces requirements have been successfully validated and verified (including technical budgets and code coverage).
- Release the software release document.
- Assess the acceptance facilities configuration.

- Verify that the Software Configuration Item under review is a formal version under configuration control.
- Evaluate known unresolved issues that can have major impact and resolution plan identification.
- Check the status of the ISVV activities.
- Check the quality assurance reports.
- Evaluate the security verification and validation activities.
- Evaluate readiness to proceed to the next phase.
- Verify the maintenance plan.
- Check the acceptance test plan.

NOTE 11Typical objectives of the software acceptance review (AR) are:

- Assess the acceptance test results, including as-run procedures.
- Verify that the Software Configuration Item under review is a formal version under configuration control
- Verify that all the RB software requirements have been successfully validated and verified (including technical budgets and code coverage) throughout the development life cycle.
- Check the status of the ISVV activities.
- Check the software acceptance data package.
- Verify that the complete set of acceptance test cases is run on the same software version.
- Accept the software product.
- Verify the software release document, the installation procedure and report and the maintenance plan.
- Evaluate known unresolved issues that can have major impact and identification of resolution plan for each outstanding issue and known problems.
- Correct closure of major SPRs/NCRs.
- Assess the RFWs.
- Check the quality assurance reports.

<7> Expected results

ECSS-E-ST-40_0860656

a. The SRevP shall include:

1. The review entry criteria:

   (a) Review data package is ready.

   (b) Review team, organization and plan are agreed and ready.

2. Review success criteria:

   (a) Review objectives are met.

   (b) Actions agreed to be closed before this review have been closed.

   (c) RIDs agreed to be closed before this review have been closed.

   (d) RIDs from this review are dispositioned and actions assigned.

3. The review conclusion, based on the review success criteria.

4. The review report, including review minutes of meeting content, presentations, RID status metric, dispositioned RIDs, discussions, actions, review conclusion and any other material used during the review.

NOTE To item 3: Possible review conclusions are:

- Successful: review success criteria have been met. Authorization to proceed with next life cycle phase is granted.
- Successful with rework to be done: review success criteria have been partially met. There are pending corrections for documents and/or Software. Corrections are done according to open SPRs and actions agreed on RIDs. Authorization to proceed with next life cycle phase is granted. Dates for closure of open SPRs and implementation of actions from RIDs are agreed in the review.
- Not successful: review success criteria have not been met. Documents to be baselined at the review cannot be baselined, and Software released for the review cannot be used for follow on activities in their current status. Authorization to proceed with next life cycle phase is not granted.

<8> Review process

ECSS-E-ST-40_0860657

a. The complete review process shall be defined, including:

1. Review planning;

2. Review participants invitation and confirmation;

3. Kick-off meeting (KOM);
4. Review datapackage(s) readiness check (optional);
5. Review datapackage(s) presentation to participants (optional);
6. Review datapackage(s) distribution;
7. Review group study of documents/deliveries followed by the generation of RIDs;
8. Review RIDs proposed disposition;
9. Review meeting(s): e.g. Review group/supplier meetings, Review group closing meeting, Decision making authority meeting;
10. Review actions closure;
11. Review closure.

ECSS-E-ST-40_0860716

b. The agenda of the presentation session may be as follows:

1. Presentation of the review group and its report;
2. Presentation of the project (context, technical and management requirements);
3. Presentation of the product (definition, critical points, performance, operations);
4. State of recommendations of the previous review (if any).

<9> Review schedule

ECSS-E-ST-40_0860659

a. The SRevP shall include a description of activity flow from data package delivery up to and including review group meeting and sequential dates.

<10> Documentation subject to review

ECSS-E-ST-40_0860660

a. The SRevP shall include:

1. The list of documents and deliveries (not only documents) subject for review;
2. Reference and applicable documents for the review;
3. The review datapackage description, including the dependencies of its deliveries subject for review.

<11> Participants

ECSS-E-ST-40_0860661

a. The SRevP shall include, as per ECSS-M-ST-10-01 clause 5.3:

1. Decision making authority responsibilities;
2. Review group chairperson, secretary and members of the review group, the supplier project team, and their responsibilities;
3. Level of independence of members.

<12> Logistics

ECSS-E-ST-40_0860662

a. The SRevP shall include:

1. The exact review address;
2. Instructions on how to arrive to the meeting location and how to get to the meeting room, including security checks needs;
3. Other logistic needs such as: LCD projector, room size, beverages available, and layout of the meeting room;
4. Suggestions on the nearest or more suitable accommodation possibilities;
5. A local point of contact, including name of contact person and complete address and phone numbers and e-mail.

<13> Annex - RID form

ECSS-E-ST-40_0860663

a. The SRevP shall include a RID form in conformance with ECSS-M-ST-10-01.

P.2.2 Special remarks

None.

Annex Q (informative) Document organization and contents at each review

Q.1 Introduction

The following clauses list the software items per review with the following columns:

- the DRD, where “-” means that there is no DRD and “ blank” means that it is an immaterial output,
- the requirement number and expected output number if needed,
- the name of the expected output,
- the trace into the DRD,
- the file.

The list is sorted per file, then per DRD, then per requirement number.

Q.2 SRR

Table Q-1: Documents content at SRR

DRD	Requirement	Expected output	Name of expected output	Trace to DRD	File
SSS	5.2.2.1a	a	Functions and performance system requirements allocated to software	<5.2>	RB
SSS	5.2.2.1a	b	Verification and validation product requirements	<6.3>, <6.4>	RB
SSS	5.2.2.1a	c	Software operations requirements	<5.11>	RB
SSS	5.2.2.1a	d	Software maintenance requirements	<5.12>	RB
SSS	5.2.2.1a	e	Requirements for in flight modification capabilities	<5.12>	RB
SSS	5.2.2.1a	f	Requirements for real- time	<5.2>3	RB
SSS	5.2.2.1a	g	Requirements for security	<5.6>	RB
SSS	5.2.2.1a	h	Quality requirements	<5.9>	RB
SSS	5.2.2.2a		System and software observability requirements	<5.13>	RB
SSS	5.2.2.3a		HMI requirements	<5.2>	RB
SSS	5.2.2.4	a	Hardware/Software co-engineering functional requirements	<5.3>b.2	RB

DRD	Requirement	Expected output	Name of expected output	Trace to DRD	File
SSS	5.2.2.4	c	Hardware/Software co-engineering design and development constraints	<5.10>b.4	RB
SSS	5.2.3.1a		Verification and validation process requirements	<6.1>a	RB
SSS	5.2.3.1b		Verification and validation security process requirements	<6.1>b	RB
SSS	5.2.3.2a		Validation requirements and scenario	<6.2>	RB
SSS	5.2.3.3a		Installation an acceptance requirements at the operational and maintenance sites	<6.4>a.1	RB
SSS	5.2.3.3b		Installation an acceptance requirements at the operational and maintenance sites	<6.4>a.1	RB
SSS	5.2.4.1a		Association of requirements to versions	<6.4>a.2	RB
SSS	5.2.4.1b		Delivery content and media	<6.4>a.2	RB
SSS	5.2.4.2a		System level integration support requirements	<6.4>a.3	RB
SSS	5.2.4.4a		System database content and allowed operational range	<5.4>	RB
SSS	5.2.4.5a		Design and development constraints	<5.10>	RB
SSS	5.2.4.6a		OBCP requirements	<5.2>e	RB
SSS	5.2.4.7a		Requirements for ’software to be reused’	<5.9>	RB
SSS	5.2.4.8a		Software safety and dependability requirements	<5.7>, <5.8>	RB
SSS	5.2.4.9a		Format and delivery medium of exchanged data	<6.4>a.4	RB
SSS	5.2.4.10a		Security constraints for the development and integration environments	<5.14>	RB
SSS	5.2.4.11a		Secure software delivery requirements	<5.15>	RB
SSS	5.3.8.1a		Technical budgets and margin philosophy for the project	<5.5>	RB
IRD	5.2.4.3a		External interface requirements specification	All	RB
IRD	5.2.4.3b		External interface requirements specification	All	RB
	5.3.4.1a		Approved requirements baseline		RB
SRevP	5.3.3.2b		Review Plan	All	MGT
SDP	5.3.2.1a		Software life cycle definition	<5.2.1>	MGT
SDP	5.3.2.1b		Software life cycle definition	<5.2.1>	MGT
SDP	5.3.2.1d		Software life cycle definition	<5.2.3>, <5.5>	MGT

DRD	Requirement	Expected output	Name of expected output	Trace to DRD	File
SDP	5.3.2.3a		Software procurement process documentation and implementation	<4.8>	MGT
SDP	5.3.2.4a		Autocode input model review	<5.2>	MGT
SDP	5.3.2.4b		Autocode interface definition and resource allocation	<5.3>	MGT
SDP	5.3.2.4c		Automatic code generation development process and tools	<5.3>	MGT
SDP	5.3.2.4d		Automatic code generation verification and validation strategy	<5.4>	MGT
SDP	5.3.3.2a		Software project reviews included in the software life cycle definition	<5.2.3>	MGT
SDP	5.3.3.3a		Software technical reviews included in the software life cycle definition	<5.2.3>	MGT
SDP	5.3.3.3b		Technical reviews process	<5.2.3>	MGT
SDP	5.3.3.3c		Software technical reviews included in the software life cycle definition	<5.2.3>	MGT
SDP	5.3.6.1a		Flight software review phasing	<5.2.2>	MGT
SDP	5.3.6.1b		Flight software review phasing	<5.2.2>	MGT
SDP	5.3.6.2a		Ground software review phasing	<5.2.2>	MGT
SDP	5.3.9.1a		ECSS-E-ST-40 compliance matrix	<5.6>	MGT
SDP	5.3.9.2a		ECSS-E-ST-40 compliance matrix	<5.6>	MGT
SCMP	5.3.2.4e		Automatic code generation configuration management		MGT
SCMP	5.3.2.5a		Changes to baselines		MGT
-	5.3.7.1a		Interface management procedures		MGT
SVR	5.3.8.2a		Technical budgets and margin computation	<5.2>	DJF
SVR	5.3.8.2b		Technical budgets and margin computation	<5.2>	DJF
SVR	5.8.3.1a		Requirements baseline verification report	<4.2>	DJF
-	5.3.3.1a		Joint review reports		DJF

Q.3 PDR

Q.3.1 PDR/SWRR

Table Q-2: Documents content at PDR/SWRR

DRD	Requirement	Expected output	Name of expected output	Trace to DRD	File
SRS	5.4.2.1a	a	Functional and performance specifications, including hardware characteristics, and environmental conditions under which the software item executes, including budgets requirements	<4.2>, <5.2>, <5.3>, <5.6>	TS
SRS	5.4.2.1a	b	Operational, reliability, safety, maintainability, portability, configuration, delivery, adaptation and installation requirements, design constraints	<5.5>, <5.2>, <5.9>, <5.11>, <5.12>, <5.13>, <5.14>, <5.17>	TS
SRS	5.4.2.1a	c	Software product quality requirements	<5.10>	TS
SRS	5.4.2.1a	d	Security specifications, including those related to factors which can compromise sensitive information	<5.8>	TS
SRS	5.4.2.1a	e	Human factors engineering (ergonomics) specifications, including those related to manual operations, human-equipment interactions, constraints on personnel, and areas requiring concentrated human attention, that are sensitive to human errors and training	<5.16>	TS
SRS	5.4.2.1a	f	Data definition and database requirements	<5.15>	TS
SRS	5.4.2.1a	g	Validation requirements	<6>	TS
SRS	5.4.2.1a	i	Reuse requirements	<5.7>b.7	TS
SRS	5.4.2.2a		Specifications for in flight software modifications	<5.7>b.5	TS
SRS	5.4.2.3a		Software logical model	<8>	TS
SRS	5.4.2.3b		Software logical model method	<8>	TS
SRS	5.4.2.3c		Behavioural view in software logical model	<8>	TS
SRS	5.8.3.2a	a	Requirements traceability matrices	<7>, <5.1>c	TS
ICD	5.4.2.1a	g	Validation requirements	<6>	TS
ICD	5.4.2.1a	h	Interfaces external to the software item	<5.2>	TS
ICD	5.8.3.2a	a	Requirements traceability matrices	<7>	TS

DRD	Requirement	Expected output	Name of expected output	Trace to DRD	File
	5.3.4.2b		Approved technical specification and interface		TS
SVR	5.8.3.12a		Technical budgets - memory and PU estimation	<5>	DJF
SVR	5.8.3.13a		Software behaviour verification	<4.3.2>	DJF
SVR	5.8.3.2a	a	Requirements traceability matrices	<4.3>1	DJF
SVR	5.8.3.2a	b	Requirements verification report	<4.3>2	DJF

Q.3.2 PDR (in addition to PDR/SWRR)

Table Q-3: Documents content at PDR (in addition to PDR/SWRR)

DRD	Requirement	Expected output	Name of expected output	Trace to DRD	File
ICD	5.4.3.5a	a	Preliminary external interfaces design	<5.3>	TS
	5.3.4.2a		Approved technical specification and interface, architecture and plans		TS
SRevP	5.3.3.2b		Review Plan	All	MGT
SDP	5.3.2.1a		Software life cycle definition	<5.2.1>	MGT
SDP	5.3.2.1b		Software life cycle definition	<5.2.1>	MGT
SDP	5.3.2.1c		Development strategy, standards, techniques, development and testing environments	<<5.4>5.1>, <5.3>,	MGT
SDP	5.3.2.1d		Software life cycle definition	<5.2.3>, <5.5>	MGT
SDP	5.3.2.2a		Identification of interface between development and maintenance	<5.2.1>	MGT
SDP	5.3.2.3a		Software procurement process documentation and implementation	<4.8>	MGT
SDP	5.3.2.4a		Autocode input model review	<5.2>	MGT
SDP	5.3.2.4b		Autocode interface definition and resource allocation	<5.3>	MGT
SDP	5.3.2.4c		Automatic code generation development process and tools	<5.3>	MGT
SDP	5.3.2.4d		Automatic code generation verification and validation strategy	<5.4>	MGT
SDP	5.3.3.2a		Software project reviews included in the software life cycle definition	<5.2.3>	MGT
SDP	5.3.3.3a		Software technical reviews included in the software life cycle definition	<5.2.3>	MGT
SDP	5.3.3.3b		Technical reviews process	<5.2.3>	MGT

DRD	Requirement	Expected output	Name of expected output	Trace to DRD	File
SDP	5.3.3.3c		Software technical reviews included in the software life cycle definition	<5.2.3>	MGT
SDP	5.3.6.1a		Flight software review phasing	<5.2.2>	MGT
SDP	5.3.6.1b		Flight software review phasing	<5.2.2>	MGT
SDP	5.3.6.2a		Ground software review phasing	<5.2.2>	MGT
SDP	5.3.9.1a		ECSS-E-ST-40 compliance matrix	<5.6>	MGT
SDP	5.3.9.2a		ECSS-E-ST-40 compliance matrix	<5.6>	MGT
SCMP	5.3.2.4e		Automatic code generation configuration management		MGT
SCMP	5.3.2.5a		Changes to baselines procedures		MGT
	5.3.4.2a		Approved technical specification and interface, architecture and plans		MGT
SVR	5.3.8.2a		Technical budgets and margin computation	<5.2>	DJF
SVR	5.3.8.2b		Technical budgets and margin computation	<5.2>	DJF
SVR	5.8.3.10a		Software documentation verification report	<4.3.2>, <4.3>a.2, <4.4>a.2, <4.5>a.2	DJF
SVR	5.8.3.11a		Schedulability analysis	<5>	DJF
SVR	5.8.3.13b		Software behaviour verification	<4.3.2>	DJF
SVR	5.8.3.3a	a	Software architectural design to requirements traceability matrices	<4.3>1	DJF
SVR	5.8.3.3a	b	Software architectural design and interface verification report	<4.3>2	DJF
SVerP	5.8.2.1a		Software verification plan - verification process identification[	<4>	DJF
SVerP	5.8.2.1b		Software verification plan - software products identification	<4>	DJF
SVerP	5.8.2.1c		Software verification plan - activities, methods and tools	<6>	DJF
SVerP	5.8.2.1d		Software verification plan - organizational independence, risk and effort identification	<4>	DJF
SValP	5.6.2.1a		Software validation plan - validation process identification	<4>, <6>	DJF
SValP	5.6.2.1b		Software validation plan - methods and tools	<4.6>, <5>, <7>	DJF
SValP	5.6.2.1c		Software validation plan - effort and independence	<4>	DJF

DRD	Requirement	Expected output	Name of expected output	Trace to DRD	File
SValP	5.8.3.9a		Complement of validation at system level	<9>	DJF
SUITP	5.4.3.8a		Software integration strategy	<5>, <6>, <7>	DJF
SRF	5.4.3.6a		Software intended for reuse - justification of methods and tools	<4>, <6>b.3	DJF
SRF	5.4.3.6b		Software intended for reuse - evaluation of reuse potential	<4>, <5>, <6>	DJF
SRF	5.4.3.7a		Justification of reuse with respect to requirements baseline	<4>, <5>	DJF
-	5.3.3.1a		Joint review reports		DJF
-	5.6.2.2a		Independent software validation plan - organization selection		DJF
-	5.6.2.2b		Independent software validation plan - level of independence		DJF
-	5.8.2.2a		Independent software verification plan - organization selection		DJF
-	5.8.2.2b		Independent software verification plan - level of independence		DJF
	5.3.4.2a		Approved technical specification and interface, architecture and plans		DJF
SDD	5.4.3.1a		Software architectural design	<4.1>, <4.2>, <4.3>, <5.1>,<5.2>, <5.3>	DDF
SDD	5.4.3.2a		Software architectural design method	<4.6>, <4.7>	DDF
SDD	5.4.3.3a		Computational model	<5.2>c	DDF
SDD	5.4.3.4a		Software behaviour	<4.3>, <5.2>e	DDF
SDD	5.4.3.5a	b	Preliminary internal interfaces design	<4.4>	DDF
SDD	5.4.3.6c		Software architectural design with configuration data	<4.1>c	DDF
SDD	5.8.3.3a	a	Software architectural design to requirements traceability matrices	<6>	DDF
	5.3.4.2a		Approved technical specification and interface, architecture and plans		DDF

Q.4 TRR

Table Q-4: Documents content at TRR (in addition to TRR/SVSR)

DRD	Requirement	Expected output	Name of expected output	Trace to DRD	File
	5.3.5.1a		Confirmation of readiness of test activities		DJF

Q.5 TRB

Table Q-5: Documents content at TRB

DRD	Requirement	Expected output	Name of expected output	Trace to DRD	File
	5.3.5.2a		Approved test results		DJF

Q.6 CDR

Q.6.1 CDR/DDR

Table Q-6: Documents content at CDR/DDR

DRD	Requirement	Expected output	Name of expected output	Trace to DRD	File
ICD	5.5.2.2a	a	External interfaces design (update)	<5.3>	TS
SVR	5.8.3.11b		Schedulability analysis (update)	<5>	DJF
SVR	5.8.3.12b		Technical budgets - memory and PU estimation (update)	<5>	DJF
SVR	5.8.3.13c		Software behaviour verification	<4.4>a.2	DJF
SVR	5.8.3.4a	a	Detailed design traceability matrices	<4.4>a.1	DJF
SVR	5.8.3.4a	b	Detailed design verification report	<4.4>a.2	DJF
SVR	5.8.3.6a	a	Software unit tests traceability matrices	<4.4>a.1	DJF
SUITP	5.5.2.9a		Software unit test plan	<5>, <6>, <7>, <8>, <9>	DJF
SRF	5.4.3.6b		Software intended for reuse - evaluation of reuse potential	<4>, <5>, <6>	DJF
	5.3.4.3b		Approved detailed design, interface design and budget		DJF
SUM	5.5.2.8a		Software user manual	All	DDF
SDD	5.4.3.6c		Software architectural design with configuration data	<4.1>c	DDF
SDD	5.5.2.1a		Software components design documents	<5.4>	DDF

DRD	Requirement	Expected output	Name of expected output	Trace to DRD	File
SDD	5.5.2.1b		Software components design documents	<5.4>	DDF
SDD	5.5.2.1c		Software components design documents	<5.4>	DDF
SDD	5.5.2.2a	b	Internal interfaces design (update)	<5.5>	DDF
SDD	5.5.2.3a	a	Software static design model	<5.4>	DDF
SDD	5.5.2.3a	b	Software dynamic design model	<5.4>	DDF
SDD	5.5.2.3a	c	Software behavioural design model	<5.4>	DDF
SDD	5.5.2.4a		Software design method	<4.7>	DDF
SDD	5.5.2.5a		Real-time software dynamic design model	<5.2>c	DDF
SDD	5.5.2.5b		Real-time software dynamic design model	<5.2>c	DDF
SDD	5.5.2.5c		Real-time software dynamic design model	<5.2>c	DDF
SDD	5.5.2.5d		Real-time software dynamic design model	<5.2>c	DDF
SDD	5.5.2.5e		Real-time software dynamic design model	<5.2>c	DDF
SDD	5.5.2.6a		Software behavioural design model techniques	<4.7>	DDF
SDD	5.5.2.7a		Compatibility of real-time design methods with the computational model	<4.7>	DDF
SDD	5.8.3.4a	a	Detailed design traceability matrices	<6>	DDF
	5.3.4.3b		Approved detailed design, interface design and budget		DDF

Q.6.2 CDR (in addition to CDR/DDR)

Table Q-7: Documents content at CDR (in addition to CDR/DDR)

DRD	Requirement	Expected output	Name of expected output	Trace to DRD	File
SVS	5.6.3.1a		Software validation specification with respect to the technical specification	<4>, <5>, <6>, <7>, <8>, <10>, <11>	DJF
SVS	5.6.3.1b		Software validation specification with respect to the technical specification	<4>, <5>, <6>,<7>, <8>, <10>,<11>	DJF

DRD	Requirement	Expected output	Name of expected output	Trace to DRD	File
SVS	5.6.3.1c		Software validation specification with respect to the technical specification	<5>, <9>	DJF
SVS	5.8.3.8a	b	Traceability of the technical specification to the validation specification	<11>	DJF
SVR	5.8.3.10a		Software documentation verification report	<4.3.2>, <4.3>a.2, <4.4>a.2, <4.5>a.2	DJF
SVR	5.8.3.11c		Schedulability analysis (update)	<5>	DJF
SVR	5.8.3.12c		Technical budgets - memory and PU estimation (update)	<5>	DJF
SVR	5.8.3.5a	a	Software code traceability matrices	<4.4>a.1	DJF
SVR	5.8.3.5a	b	Software code verification report	<4.4>a.2	DJF
SVR	5.8.3.5b		Code coverage verification report	<4.4>a.2, <4.5>a.2	DJF
SVR	5.8.3.5c		Code coverage verification report	<4.4>a.2, <4.5>a.2	DJF
SVR	5.8.3.5d		Code coverage verification report	<4.4>a.2, <4.5>a.2	DJF
SVR	5.8.3.5e		Code coverage verification report	<4.4>a.2, <4.5>a.2	DJF
SVR	5.8.3.5f		Robustness verification report	<4.4>a.2	DJF
SVR	5.8.3.6a	b	Software unit testing verification report	<4.4>a.2	DJF
SVR	5.8.3.7a		Software integration verification report	<4.4>a.2	DJF
SVR	5.8.3.8a	b	Traceability of the technical specification to the validation specification	<4.6>a.1	DJF
SVR	5.8.3.8b	a	Validation report evaluation with respect to the technical specification	<4.6>a.2	DJF
SUITP	5.5.3.2a	b	Software unit test plan (update)	<10>, <11>	DJF
SUITP	5.5.4.1a		Software integration test plan (update)	<8>, <9>, <10>, <11>	DJF
-	5.3.3.1a		Joint review reports		DJF
-	5.5.3.2b	b	Software unit test report		DJF
-	5.5.3.2c		Software unit test report		DJF
-	5.5.4.2a		Software integration test report		DJF
-	5.6.3.2a		Software validation report with respect to the technical specification		DJF
	5.3.4.3a		Approved design definition file and design justification file		DJF
SUM	5.6.3.3a		Software user manual (update)	All	DDF

DRD	Requirement	Expected output	Name of expected output	Trace to DRD	File
source	5.5.3.1a	a	Software component design documents and code (update)		DDF
source	5.5.3.2a	a	Software component design documents and code (update)		DDF
source	5.5.3.2b	a	Software component design document and code (update)		DDF
SDD	5.5.3.1a	a	Software component design documents and code (update)	<5>	DDF
SDD	5.5.3.2a	a	Software component design documents and code (update)	<5>	DDF
SDD	5.5.3.2b	a	Software component design document and code (update)	<5>	DDF
SCF	5.5.3.1a	b	Software configuration file - build procedures		DDF
	5.3.4.3a		Approved design definition file and design justification file		DDF

Q.7 QR

Table Q-8: Documents content at QR

DRD	Requirement	Expected output	Name of expected output	Trace to DRD	File
	5.3.4.4a		Qualified software product		TS
	5.3.4.4a		Qualified software product		RB
	5.3.4.4a		Qualified software product		MGT
-	5.10.2.1a		Maintenance plan - plans and procedures		MF
-	5.10.2.1b		Maintenance plan - applicability of development process procedures, methods, tools and standards		MF
-	5.10.2.1c		Maintenance plan - configuration management process		MF
-	5.10.2.1d		Maintenance plan - problem reporting and handling		MF
-	5.10.2.1e		Problem and nonconformance report		MF
-	5.10.2.2a		Maintenance plan - long term maintenance solutions		MF
	5.3.4.4a		Qualified software product		MF
SVS	5.6.4.1a		Software validation specification with respect to the requirements baseline	<4>, <5>, <6>, <7>, <8>, <10>, <11>	DJF

DRD	Requirement	Expected output	Name of expected output	Trace to DRD	File
SVS	5.6.4.1b		Software validation specification with respect to the requirements baseline	<4>, <5>, <6>, <7>, <8>, <10>, <11>	DJF
SVS	5.6.4.1c		Software validation specification with respect to the requirements baseline	<5>, <9>	DJF
SVS	5.8.3.8a	a	Traceability of the requirements baseline to the validation specification	<11>	DJF
SVR	5.8.3.10a		Software documentation verification report	<4.3.2>, <4.3>a.2, <4.4>a.2, <4.5>a.2	DJF
SVR	5.8.3.11d		Schedulability analysis (update)	<5>	DJF
SVR	5.8.3.12c		Technical budgets - memory and PU estimation (update)	<5>	DJF
SVR	5.8.3.5b		Code coverage verification report	<4.5>a.2	DJF
SVR	5.8.3.5c		Code coverage verification report	<4.4>a.2, <4.5>a.2	DJF
SVR	5.8.3.5d		Code coverage verification report	<4.4>a.2, <4.5>a.2	DJF
SVR	5.8.3.5e		Code coverage verification report	<4.4>a.2, <4.5>a.2	DJF
SVR	5.8.3.8a	a	Traceability of the requirements baseline to the validation specification	<4.6>a.1	DJF
SVR	5.8.3.8b	b	Validation report evaluation with respect to the requirements baseline	<4.6>a.2	DJF
-	5.3.3.1a		Joint review reports		DJF
-	5.6.4.2a		Software validation report with respect to the requirements baseline		DJF
-	5.6.4.2b		Software validation report with respect to the requirements baseline		DJF
-	5.7.3.1a		Acceptance test plan		DJF
	5.3.4.4a		Qualified software product		DJF
SUM	5.6.4.3a		Software user manual (update)	All	DDF
Srel	5.7.2.1a	b	Software release document	All	DDF
-	5.7.2.1a	a	Software product		DDF
-	5.7.2.2a		Training material		DDF
	5.3.4.4a		Qualified software product		DDF

Q.8 AR

Table Q-9: Documents content at AR

DRD	Requirement	Expected output	Name of expected output	Trace to DRD	File
	5.3.4.5a		Accepted software product		TS
	5.3.4.5a		Accepted software product		RB
	5.3.4.5a		Accepted software product		MGT
-	5.10.2.1a		Maintenance plan - plans and procedures		MF
-	5.10.2.1b		Maintenance plan - applicability of development process procedures, methods, tools and standards		MF
-	5.10.2.1c		Maintenance plan - configuration management process		MF
-	5.10.2.1d		Maintenance plan - problem reporting and handling		MF
-	5.10.2.2a		Maintenance plan - long term maintenance solutions		MF
	5.3.4.5a		Accepted software product		MF
SVS	5.6.4.1a		Software validation specification with respect to the requirements baseline	<4>, <5>, <6>, <7>, <8>, <10>, <11>	DJF
SVS	5.6.4.1b		Software validation specification with respect to the requirements baseline	<4>, <5>, <6>, <7>, <8>, <10>, <11>	DJF
SVS	5.6.4.1c		Software validation specification with respect to the requirements baseline	<5>, <9>	DJF
SVS	5.8.3.8a	a	Traceability of the requirements baseline to the validation specification	<11>	DJF
SVR	5.7.3.5a		Traceability of acceptance tests to the requirements baseline	<4.5>a.1	DJF
SVR	5.8.3.12c		Technical budgets - memory and PU estimation (update)	<5>	DJF
SVR	5.8.3.5b		Code coverage verification report	<4.4>a.2, <4.5>a.2	DJF
SVR	5.8.3.5c		Code coverage verification report	<4.4>a.2, <4.5>a.2	DJF
SVR	5.8.3.5d		Code coverage verification report	<4.4>a.2,	
SVR	5.8.3.5e		Code coverage verification report	<4.4>a.2, <4.5>a.2	DJF

DRD	Requirement	Expected output	Name of expected output	Trace to DRD	File
SVR	5.8.3.8a	a	Traceability of the requirements baseline to the validation specification	<4.6>a.1	DJF
-	5.3.3.1a		Joint review reports		DJF
-	5.6.4.2a		Software validation report with respect to the requirements baseline		DJF
-	5.6.4.2b		Software validation report with respect to the requirements baseline		DJF
-	5.7.2.4a		Installation report		DJF
-	5.7.2.4b		Installation report		DJF
-	5.7.2.4c		Installation report		DJF
-	5.7.2.4d		Installation report		DJF
-	5.7.3.1a		Acceptance test plan		DJF
-	5.7.3.2a		Acceptance test report		DJF
-	5.7.3.4a		Joint review reports		DJF
-	5.7.3.4b		Joint review reports		DJF
	5.3.4.5a		Accepted software product		DJF
SUM	5.6.4.3a		Software user manual (update)	All	DDF
Srel	5.7.2.1a	b	Software release document	All	DDF
SCF	5.7.2.3a		Installation procedure		DDF
-	5.7.2.1a	a	Software product		DDF
-	5.7.3.3a		Software product		DDF
	5.3.4.5a		Accepted software product		DDF

Q.9 ORR

Table Q-10: Documents content at ORR

DRD	Requirement	Expected output	Name of expected output	Trace to DRD	File
-	5.9.2.1a		Software operation support plan - operational testing specifications		OP
-	5.9.2.2a		Software operation support plan - plans and procedures		OP
-	5.9.2.3a		Software operation support plan - procedures for problem handling		OP
-	5.9.3.1a		Operational testing results		OP
-	5.9.3.2a		Operational testing results		OP
-	5.10.2.1a		Maintenance plan - plans and procedures		MF

-	5.10.2.1b		Maintenance plan - applicability of development process procedures, methods, tools and standards		MF
-	5.10.2.1c		Maintenance plan - configuration management process		MF
-	5.10.2.1d		Maintenance plan - problem reporting and handling		MF
-	5.10.2.2a		Maintenance plan - long term maintenance solutions		MF
-	5.9.3.3a		Software product		DDF

Q.10 No explicit review

Table Q-11: Documents content of documents with no explicit review

DRD	Requirement	Expected output	Name of expected output	Trace to DRD	File
-	5.10.6.6a		Post operation review report		OP
-	5.10.6.6b		Post operation review report		OP
-	5.9.4.2a		Problem and nonconformance report		OP
-	5.9.5.1a		User’s requests record - user’s request and subsequent actions		OP
-	5.9.5.1b		User’s requests record - user’s request and subsequent actions		OP
-	5.9.5.2a		User’s requests record - actions		OP
-	5.9.5.2b		User’s requests record - actions		OP
-	5.9.5.2c		User’s requests record - actions		OP
-	5.9.5.3a		User’s requests record - work-around solutions		OP
-	5.9.5.3b		User’s requests record - work-around solutions		OP
-	5.10.3.1a		Modification analysis report and problem analysis report		MF
-	5.10.3.1b		Modification analysis report and problem analysis report		MF
-	5.10.3.1c		Modification analysis report and problem analysis report		MF
-	5.10.3.1d		Modification analysis report and problem analysis report		MF
-	5.10.4.1a		Modification documentation		MF
-	5.10.4.2a		Modification documentation		MF
-	5.10.4.3a		Modification documentation		MF
-	5.10.4.3b		Modification documentation		MF

DRD	Requirement	Expected output	Name of expected output	Trace to DRD	File
-	5.10.4.3c		Modification documentation		MF
-	5.10.4.3d		Modification documentation		MF
-	5.10.4.3e		Modification documentation		MF
-	5.10.5.1a		Joint review reports		MF
-	5.10.5.2a		Baseline for changes		MF
-	5.10.6.1a		Migration plan		MF
-	5.10.6.2a		Migration plan		MF
-	5.10.6.3a		Migration plan		MF
-	5.10.6.4a		Migration plan		MF
-	5.10.6.5a		Migration notification		MF
-	5.10.6.5b		Migration notification		MF
-	5.10.6.7a		Migration plan		MF
-	5.10.7.1a		Retirement plan		MF
-	5.10.7.2a		Retirement notification		MF
-	5.10.7.3a		Retirement plan		MF
-	5.10.7.4a		Retirement plan		MF
	5.10.3.1e		Modification approval		MF

Q.11 SW DRB

Table Q-12: Documents content at SW DRB

DRD	Requirement	Expected output	Name of expected output	Trace to DRD	File
-	5.3.5.3a		Acceptance of the software delivery		DJF

Q.12 SVSR

Table Q-13: Documents content at SVSR

DRD	Requirement	Expected output	Name of expected output	Trace to DRD	File
-	5.3.5.4a		Approved software validation specification		DJF

Annex R (normative) Tailoring of this Standard based on software criticality

R.1 Overview

The following applicability matrix represents a tailoring of the requirements of this Standard based on the software criticality categories defined in ECSS-Q-ST-80 Annex D.1.

The electronic format of a tool database can be required or agreed by the customer for any criticality category.

For each clause of this Standard and for each software criticality category, the following indication is given:

- Y: means that the requirement is applicable for that criticality category. The activity and the expected output are required;
- N: means that the requirement is not applicable for that criticality category. Neither the activity nor the expected output are required;
- Ytba: means that some DRD information may be missing if justified and agreed by the customer;
- specific conditions.

R.2 Tailoring

ECSS-E-ST-40_0860664

a. For tailoring of this standard based on software criticality categories, Table R-1 shall be applied.

ECSS-E-ST-40_0860665

Table R-1: Criticality applicability

Requirement identification	Expected Output	A	B	C	D
5.2.2.1a	Specification of system requirements allocated to software	Y	Y	Y	Y
5.2.2.1a eo a	Functions and performance system requirements allocated to software	Y	Y	Y	Y
5.2.2.1a eo b	Verification and validation product requirements	Y	Y	Y	Y
5.2.2.1a eo c	Software operations requirements	Y	Y	Y	Y
5.2.2.1a eo d	Software maintenance requirements	Y	Y	Y	Y

Requirement identification	Expected Output	A	B	C	D
5.2.2.1a eo e	Requirements for in flight modification capabilities	Y	Y	Y	Y
5.2.2.1a eo f	Requirements for real-time	Y	Y	Y	Y
5.2.2.1a eo g	Requirements for security	Y	Y	Y	Y
5.2.2.1a eo h	Quality requirements	Y	Y	Y	Y
5.2.2.1b	Additional security requirements	Y	Y	Y	Y
5.2.2.1c	Security requirements catalogues	Y	Y	Y	Y
5.2.2.1d	Security traceability matrix	Y	Y	Y	Y
5.2.2.2a	System and software observability requirements	Y	Y	Y	Y
5.2.2.3a	HMI requirements	Y	Y	Y	Y
5.2.2.4a	HW/SW co-engineering analysis	Y	Y	Y	Y
5.2.3.1a	Verification and validation process requirements	Y	Y	Y	Y
5.2.3.1b	Security verification and validation process requirements	Y	Y	Y	Y
5.2.3.2a	Validation requirements and scenario	Y	Y	Y	Y
5.2.3.3a	Installation an acceptance requirements at the operational and maintenance sites	Y	Y	Y	Y
5.2.3.3b	Secure installation and security specific acceptance criteria requirements	Y	Y	Y	Y
5.2.4.1a	Association of requirements to versions	Y	Y	Y	Y
5.2.4.1b	Delivery content and media	Y	Y	Y	Y
5.2.4.2a	System level integration support requirements	Y	Y	Y	Y
5.2.4.3a	External interface requirements specification	Y	Y	Y	Y
5.2.4.3b	External interface security requirements specification	Y	Y	Y	Y
5.2.4.4a	System database content and allowed operational range	Y	Y	Y	Y
5.2.4.4b	System database security constraints	Y	Y	Y	Y
5.2.4.5a	Design and development constraints	Y	Y	Y	Y
5.2.4.6a	OBCP requirements	Y	Y	Y	Y
5.2.4.7a	Requirements for ’software to be reused’	Y	Y	Y	Y

Requirement identification	Expected Output	A	B	C	D
5.2.4.8a	Software safety and dependability requirements	Y	Y	Y	Y
5.2.4.9a	Format and delivery medium of exchanged data	Y	Y	Y	Y
5.2.4.9b	Security marking and labelling requirements	Y	Y	Y	Y
5.2.4.9c	Integrity and authenticity requirements	Y	Y	Y	Y
5.2.4.10a	Security constraints for the development and integration environments	Y	Y	Y	Y
5.2.4.11a	Secure software delivery requirements	Y	Y	Y	Y
5.2.5a	SRR	Y	Y	Y	Y
5.3.2.1a	Software life cycle definition	Y	Y	Y	Y
5.3.2.1b	Software life cycle definition	Y	Y	Y	Y
5.3.2.1c	Development strategy, standards, techniques, development and testing environment	Y	Y	Y	Y
5.3.2.1d	Software life cycle definition	Y	Y	Y	Y
5.3.2.2a	Identification of interface between development and maintenance	Y	Y	Y	Y
5.3.2.3a	Software procurement process documentation and implementation	Y	Y	Y	Y
5.3.2.4a	Automatic code generation management	Y	Y	Y	Y
5.3.2.4b	Automatic code generation management	Y	Y	Y	Y
5.3.2.4c	Automatic code generation management	Y	Y	Y	Y
5.3.2.4d	Automatic code generation management	Y	Y	Y	Y
5.3.2.4e	Automatic code generation configuration management	Y	Y	Y	Y
5.3.2.5a	Changes to baseline procedures	Y	Y	Y	Y
5.3.3.1a	Joint review reports	Y	Y	Y	Y
5.3.3.1b	Security specific sessions	Y	Y	Y	Y
5.3.3.2a	Software project reviews included in the software life cycle definition	Y	Y	Y	Y
5.3.3.2b	Review Plan	Y	Y	Y	Y
5.3.3.3a	Software technical reviews included in the software life cycle definition	Y	Y	Y	Y

Requirement identification	Expected Output	A	B	C	D
5.3.3.3b	Technical reviews process	Y	Y	Y	Y
5.3.3.3c	Software technical reviews included in the software life cycle definition	Y	Y	Y	Y
5.3.4.1a	Approved requirements baseline	Y	Y	Y	Y
5.3.4.2a	Approved technical specification and interface, architecture and plans	Y	Y	Y	Y
5.3.4.2b	Approved technical specification and interface	Y	Y	Y	Y
5.3.4.3a	Approved design definition file and design justification file	Y	Y	Y	Y
5.3.4.3b	Approved detailed design, interface design and budget	Y	Y	Y	Y
5.3.4.4a	Qualified software product	Y	Y	Y	Y
5.3.4.5a	Accepted software product	Y	Y	Y	Y
5.3.5.1a	Confirmation of readiness of test activities For validation and acceptance test activities only	Y	Y	Y	Applicable to validation and acceptance tests only
5.3.5.2a	Approved test results For validation and acceptance test activities only	Y	Y	Y	Applicable to validation and acceptance tests only
5.3.5.3a	Software delivery review board	Y	Y	Y	Y
5.3.5.4a	Software validation specification review	Y	Y	Y	Y
5.3.6.1a	Flight software review phasing	Y	Y	Y	Y
5.3.6.1b	Flight software review phasing	Y	Y	Y	Y
5.3.6.2a	Ground software review phasing	Y	Y	Y	Y
5.3.7.1a	Interface management procedures	Y	Y	Y	Y
5.3.8.1a	Technical budgets and margin philosophy for the project	Y	Y	Y	Y
5.3.8.2a	Technical budgets and margin computation	Y	Y	Y	Y
5.3.8.2b	Technical budgets and margin computation	Y	Y	Y	Y
5.3.9.1a	E40 compliance matrix	Y	Y	Y	Y
5.3.9.2a	E40 compliance matrix	Y	Y	Y	Y
5.4.2.1a eo a	Functional and performance specifications, including hardware characteristics, and environmental conditions under which the software	Y	Y	Y	Y

Requirement identification	Expected Output	A	B	C	D
	item executes, including budgets requirements				
5.4.2.1a eo b	Operational, reliability, safety, maintainability, portability, configuration, delivery, adaptation and installation requirements, design constraints	Y	Y	Y	Y
5.4.2.1a eo c	Software product quality requirements (see ECSS-Q-ST-80 clause 7.2)	Y	Y	Y	Y
5.4.2.1a eo d	Security specifications, including those related to factors which can compromise sensitive information	Y	Y	Y	Y
5.4.2.1a eo e	Human factors engineering (ergonomics) specifications, following the human factor engineering process described in ECSS-E-ST-10-11	Y	Y	Y	Y
5.4.2.1a eo f	Data definition and database requirements	Y	Y	Y	Y
5.4.2.1a eo g	Validation requirements	Y	Y	Y	Y
5.4.2.1a eo h	Interfaces external to the software item	Y	Y	Y	Y
5.4.2.1a eo i	Reuse requirements	Y	Y	Y	Y
5.4.2.2a	Specifications for in flight software modifications	Y	Y	Y	Y
5.4.2.3a	Software logical model	Y	Y	N	N
5.4.2.3b	Software logical model method	Y	Y	N	N
5.4.2.3c	Behavioural view in software logical model	Y	Y	N	N
5.4.2.4a	SWRR	Y	Y	Y	Y
5.4.3.1a	Software architectural design	Y	Y	Y	Y
5.4.3.2a	Software architectural design method	Y	Y	Y	Y
5.4.3.2b	Vulnerability analysis of the design	Y	Y	Y	Y
5.4.3.3a	Computational model	Y	Y	Y	Y
5.4.3.4a	Software behaviour	Y	Y	N	N
5.4.3.5a eo a	Preliminary external interfaces design	Y	Y	Y	Y
5.4.3.5a eo b	Preliminary internal interfaces design	Y	Y	Y	Y
5.4.3.6a	Software intended for reuse - justification of methods and tools	Y	Y	Y	Y
5.4.3.6b	Software intended for reuse - evaluation of reuse potential	Y	Y	Y	Y
5.4.3.6c	Software architectural design with configuration data	Y	Y	Y	Y

Requirement identification	Expected Output	A	B	C	D
5.4.3.7a	Justification of reuse with respect to requirements baseline	Y	Y	Y	Y
5.4.3.8a	Software integration strategy	Y	Y	Y	N
5.4.4a	PDR	Y	Y	Y	Y
5.5.2.1a	Software components design documents	Y	Y	Ytba	Ytba
5.5.2.1b	Software components design documents	Y	Y	Ytba	Ytba
5.5.2.1c	Software components design documents	Y	Y	Ytba	Ytba
5.5.2.2a eo a	External interfaces design (update)	Y	Y	Y	Y
5.5.2.2a eo b	Internal interfaces design (update)	Y	Y	Ytba	Ytba
5.5.2.3a eo a	Software static design model	Y	Y	Ytba	Ytba
5.5.2.3a eo b	Software dynamic design model	Y	Y	Ytba	Ytba
5.5.2.3a eo c	Software behavioural design model	Y	Y	Ytba	Ytba
5.5.2.4a	Software design method	Y	Y	Y	Y
5.5.2.5a	Real-time software dynamic design model	Y	Y	Ytba	Ytba
5.5.2.5b	Real-time software dynamic design model	Y	Y	Ytba	Ytba
5.5.2.5c	Real-time software dynamic design model	Y	Y	Ytba	Ytba
5.5.2.5d	Real-time software dynamic design model	Y	Y	Ytba	Ytba
5.5.2.5e	Real-time software dynamic design model	Y	Y	Ytba	Ytba
5.5.2.6a	Software behavioural design model techniques	Y	Y	Ytba	Ytba
5.5.2.7a	Compatibility of real-time design methods with the computational model	Y	Y	Y	Y
5.5.2.8a	Software user manual	Y	Y	Y	Y
5.5.2.9a	Software unit test plan	Y	Y	Y except Annex K sections 9 and 10	Y except Annex K sections 9, 10 and 11
5.5.2.10a	DDR	Y	Y	Y	Y
5.5.3.1a eo a	Software component design documents and code (update)	Y	Y	Ytba	Ytba
5.5.3.1a eo b	Software configuration file - build procedures	Y	Y	Y	Y

Requirement identification	Expected Output	A	B	C	D
5.5.3.2a eo a	Software component design document and code (update)	Y	Y	Ytba	Ytba
5.5.3.2a eo b	Software unit test plan (update)	Y	Y	Y	Y
5.5.3.2b eo a	Software component design document and code (update)	Y	Y	Ytba	Ytba
5.5.3.2b eo b	Software unit test reports	Y	Y	Ytba	Ytba
5.5.3.2c	Software unit test reports	Y	Y	Ytba	Ytba
5.5.4.1a	Software integration test plan (update)	Y	Y	Y except Annex K sections 9 and 10	N
5.5.4.2a	Software integration test report	Y	Y	Y	N
5.6.2.1a	Software validation plan - validation process identification	Y	Y	Y	Y
5.6.2.1b	Software validation plan - methods and tools	Y	Y	Y	Y
5.6.2.1c	Software validation plan - effort and independence	Y	Y	Y	Y
5.6.2.1d	Software validation regulations and policies compliance	Y	Y	Y	Y
5.6.2.2a	Independent software validation plan - organization selection	Y	Y	N	N
5.6.2.2b	Independent software validation plan - level of independence	Y	Y	N	N
5.6.3.1a	Software validation specification with respect to the technical specification	Y	Y	Y	Y
5.6.3.1b	Software validation specification with respect to the technical specification	Y	Y	Y	Y
5.6.3.1c	Software validation specification with respect to the technical specification	Y	Y	Y	Y
5.6.3.2a	Software validation report with respect to the technical specification	Y	Y	Y	Y
5.6.3.3a	Software user manual (update)	Y	Y	Y	Y
5.6.3.4a	CDR	Y	Y	Y	Y
5.6.4.1a	Software validation specification with respect to the requirements baseline	Y	Y	Y	Y
5.6.4.1b	Software validation specification with respect to the requirements baseline	Y	Y	Y	Y
5.6.4.1c	Software validation specification with respect to the requirements baseline	Y	Y	Y	Y

Requirement identification	Expected Output	A	B	C	D
5.6.4.2a	Software validation report with respect to the requirements baseline	Y	Y	Y	Y
5.6.4.2b	Software validation report with respect to the requirements baseline	Y	Y	Y	Y
5.6.4.3a	Software user manual (update)	Y	Y	Y	Y
5.6.4.4a	QR	Y	Y	Y	Y
5.6.5a	Software validation control	Y	Y	Y	Y
5.7.2.1a eo a	Software product	Y	Y	Y	Y
5.7.2.1a eo b	Software release document	Y	Y	Y	Y
5.7.2.1b	Secured software delivery	Y	Y	Y	Y
5.7.2.2a	Training material	Y	Y	Y	Y
5.7.2.3a	Installation procedures	Y	Y	Y	Y
5.7.2.3b	Secured installation procedures	Y	Y	Y	Y
5.7.2.4a	Installation report	Y	Y	Y	Y
5.7.2.4b	Installation report	Y	Y	Y	Y
5.7.2.4c	Installation report	Y	Y	Y	Y
5.7.2.4d	Installation report	Y	Y	Y	Y
5.7.3.1a	Acceptance test plan	Y	Y	Y	Y
5.7.3.2a	Acceptance test report	Y	Y	Y	Y
5.7.3.3a	Software product	Y	Y	Y	Y
5.7.3.4a	Joint review report	Y	Y	Y	Y
5.7.3.4b	Joint review report	Y	Y	Y	Y
5.7.3.5a	Traceability of acceptance tests to the requirements baseline	Y	Y	Y	Y
5.7.3.6a	AR	Y	Y	Y	Y
5.8.2.1a	Software verification plan - verification process identification	Y	Y	Y	Y
5.8.2.1b	Software verification plan - software products identification	Y	Y	Y	Y
5.8.2.1c	Software verification plan - activities, methods and tools	Y	Y	Y	Y
5.8.2.1d	Software verification plan - organizational independence, risk and effort identification	Y	Y	Y	Y
5.8.2.1e	Software verification regulations and policies compliance	Y	Y	Y	Y
5.8.2.2a	Independent software verification plan - organization selection	Y	Y	N	N

Requirement identification	Expected Output	A	B	C	D
5.8.2.2b	Independent software verification plan - level of independence	Y	Y	N	N
5.8.3.1a	Requirements baseline verification report	Y	Y	Y	N
5.8.3.2a eo a	Requirements traceability matrices	Y	Y	Y	Y
5.8.3.2a eo b	Requirements verification report	Y	Y	Y	N
5.8.3.3a eo a	Software architectural design to requirements traceability matrices	Y	Y	Y	Y
5.8.3.3a eo b	Software architectural design and interface verification report	Y	Y	Y	N
5.8.3.4a eo a	Detailed design traceability matrices	Y	Y	Y	N
5.8.3.4a eo b	Detailed design verification report	Y	Y	Y	N
5.8.3.5a eo a	Software code traceability matrices	Y	Y	Y	N
5.8.3.5a eo b	Software code verification report	Y	Y	Y	N
5.8.3.5b	Code coverage verification report	See 5.8.3.5b	See 5.8.3.5b	See 5.8.3.5b	See 5.8.3.5b
5.8.3.5c	Code coverage verification report	Y	Y	Y	Y
5.8.3.5d	Code coverage verification report	Y	Y	Y	Y
5.8.3.5e	Code coverage verification report	See 5.8.3.5e	See 5.8.3.5e	See 5.8.3.5e	See 5.8.3.5e
5.8.3.5f	Robustness verification report	Y	Y	Y	N
5.8.3.6a eo a	Software unit tests traceability matrices	Y	Y	N	N
5.8.3.6a eo b	Software unit testing verification report	Y	Y	N	N
5.8.3.7a	Software integration verification report	Y	Y	N	N
5.8.3.8a eo a	Traceability of the requirements baseline to the validation specification	Y	Y	Y	Y
5.8.3.8a eo b	Traceability of the technical specification to the validation specification	Y	Y	Y	Y
5.8.3.8b eo a	Validation report evaluation with respect to the technical specification	Y	Y	Y	Y
5.8.3.8b eo b	Validation report evaluation with respect to the requirements baseline	Y	Y	Y	Y
5.8.3.9a	Complement of validation at system level	Y	Y	Y	Y

Requirement identification	Expected Output	A	B	C	D
5.8.3.10a	Software documentation verification report	Y	Y	Y	N
5.8.3.11a	Schedulability analysis	Y	Y	Y	N
5.8.3.11b	Schedulability analysis (update)	Y	Y	Y	N
5.8.3.11c	Schedulability analysis (update)	Y	Y	Y	N
5.8.3.11d	Schedulability analysis environment	Y	Y	Y	N
5.8.3.12a	Technical budgets - memory and PU estimation	Y	Y	Y	Y
5.8.3.12b	Technical budgets (update) - memory and PU estimation	Y	Y	Y	Y
5.8.3.12c	Technical budgets (update) - memory and PU calculation	Y	Y	Y	Y
5.8.3.13a	Software behaviour verification	Y	Y	N	N
5.8.3.13b	Software behaviour verification	Y	Y	N	N
5.8.3.13c	Software behaviour verification	Y	Y	N	N
5.9.2.1a	Software operation support plan - operational testing specifications	Y	Y	Y	Y
5.9.2.2a	Software operation support plan - plans and procedures	Y	Y	Y	Y
5.9.2.3a	Software operation support plan - procedures for problem handling	Y	Y	Y	Y
5.9.2.3b	Software operation support plan – security impact	Y	Y	Y	Y
5.9.3.1a	Operational testing results	Y	Y	Y	Y
5.9.3.2a	Operational testing results	Y	Y	Y	Y
5.9.3.3a	Software product	Y	Y	Y	Y
5.9.4.1a	Software operation support performance	Y	Y	Y	Y
5.9.4.2a	Problem and nonconformance report	Y	Y	Y	Y
5.9.4.3a	Security vulnerabilities in operations	Y	Y	Y	Y
5.9.5.1a	User's request record - user's request and subsequent actions	Y	Y	Y	Y
5.9.5.1b	User's request record - user's request and subsequent actions	Y	Y	Y	Y
5.9.5.2a	User's request record - actions	Y	Y	Y	Y
5.9.5.2b	User's request record - actions	Y	Y	Y	Y
5.9.5.2c	User's request record - actions	Y	Y	Y	Y
5.9.5.3a	User's request record - work around solution	Y	Y	Y	Y

Requirement identification	Expected Output	A	B	C	D
5.9.5.3b	User's request record - work around solution	Y	Y	Y	Y
5.10.2.1a	Maintenance plan - plans and procedures	Y	Y	Y	Y
5.10.2.1b	Maintenance plan - applicability of development process procedures, methods, tools and standards	Y	Y	Y	Y
5.10.2.1c	Maintenance plan - configuration management process	Y	Y	Y	Y
5.10.2.1d	Maintenance plan - problem reporting and handling	Y	Y	Y	Y
5.10.2.1e	Problem and nonconformance report	Y	Y	Y	Y
5.10.2.1f	Secured software maintenance process	Y	Y	Y	Y
5.10.2.2a	Maintenance plan - long term maintenance solutions	Y	Y	Y	Y
5.10.3.1a	Modification analysis report and problem analysis report	Y	Y	Y	Y
5.10.3.1b	Modification analysis report and problem analysis report	Y	Y	Y	Y
5.10.3.1c	Modification analysis report and problem analysis report	Y	Y	Y	Y
5.10.3.1d	Modification analysis report and problem analysis report	Y	Y	Y	Y
5.10.3.1e	Modification approval	Y	Y	Y	Y
5.10.4.1a	Modification documentation	Y	Y	Y	Y
5.10.4.2a	Modification documentation	Y	Y	Y	Y
5.10.4.3a	Modification documentation	Y	Y	Y	Y
5.10.4.3b	Modification documentation	Y	Y	Y	Y
5.10.4.3c	Modification documentation	Y	Y	Y	Y
5.10.4.3d	Modification documentation	Y	Y	Y	Y
5.10.4.3e	Modification documentation	Y	Y	Y	Y
5.10.5.1a	Joint review reports	Y	Y	Y	Y
5.10.5.2a	Baseline for changes	Y	Y	Y	Y
5.10.6.1a	Migration plan	Y	Y	Y	Y
5.10.6.2a	Migration plan	Y	Y	Y	Y
5.10.6.3a	Migration plan	Y	Y	Y	Y
5.10.6.4a	Migration plan	Y	Y	Y	Y
5.10.6.5a	Migration notification	Y	Y	Y	Y
5.10.6.5b	Migration notification	Y	Y	Y	Y

Requirement identification	Expected Output	A	B	C	D
5.10.6.6a	Post operation review report	Y	Y	Y	Y
5.10.6.6b	Post operation review report	Y	Y	Y	Y
5.10.6.7a	Migration plan	Y	Y	Y	Y
5.10.7.1a	Retirement plan	Y	Y	Y	Y
5.10.7.2a	Retirement notification	Y	Y	Y	Y
5.10.7.3a	Retirement plan	Y	Y	Y	Y
5.10.7.4a	Retirement plan	Y	Y	Y	Y
5.11.2a	Software security management plan	Y	Y	Y	Y
5.11.2b	Software security management plan	Y	Y	Y	Y
5.11.2c	Software security management plan	Y	Y	Y	Y
5.11.2d	Software security management plan	Y	Y	Y	Y
5.11.3a	Software security analysis	Y	Y	Y	Y
5.11.3b	Software security analysis methods	Y	Y	Y	Y
5.11.3c	Software security analysis production	Y	Y	Y	Y
5.11.3d	Software security analysis of next higher level	Y	Y	Y	Y
5.11.3e	Software security analysis feedback	Y	Y	Y	Y
5.11.3f	Software security analysis update	Y	Y	Y	Y
5.11.3g	Software security analysis distribution	Y	Y	Y	Y
5.11.3h	Software security analysis order	Y	Y	Y	Y
5.11.3i	Software security analysis vulnerability mapping	Y	Y	Y	Y
5.11.3j	Software security analysis vulnerability mapping update	Y	Y	Y	Y
5.11.3k	Software security analysis review	Y	Y	Y	Y
5.11.3l	Software security analysis classification	Y	Y	Y	Y
5.11.4a	Security risk treatment	Y	Y	Y	Y
5.11.4b	Security risk impact	Y	Y	Y	Y
5.11.4c	Security risk protection	Y	Y	Y	Y
5.11.4d	Software requirements linked to security risk traceability	Y	Y	Y	Y
5.11.4e	Security risks, associated security risks and risk treatments consistency	Y	Y	Y	Y
5.11.4f	Risk treatment measures	Y	Y	Y	Y
5.11.5.1a	Security in the requirements baseline	Y	Y	Y	Y
5.11.5.1b	Security assurance requirements	Y	Y	Y	Y

Requirement identification	Expected Output	A	B	C	D
5.11.5.1c	Security requirement verification and validation	Y	Y	Y	Y
5.11.5.1d	Software security analysis and associated risk treatment measures review	Y	Y	Y	Y
5.11.5.2a	Security in the software requirements specification and architectural design	Y	Y	Y	Y
5.11.5.2b	Software security analysis revision	Y	Y	Y	Y
5.11.5.2c	Software security analysis and associated risk treatment plan review	Y	Y	Y	Y
5.11.5.2d	Architecture review with respect to security risks	Y	Y	Y	Y
5.11.5.3a	Security analysis review during the detailed design and implementation engineering	Y	Y	Y	Y
5.11.5.3b	Residual vulnerabilities list	Y	Y	Y	Y
5.11.5.3c	Residual vulnerabilities list review	Y	Y	Y	Y
5.11.5.3d	Additional security risks identification	Y	Y	Y	Y
5.11.5.4a	Security validation process	Y	Y	Y	Y
5.11.5.4b	Security tests results approval	Y	Y	Y	Y
5.11.5.4c	Security measures for testing	Y	Y	Y	Y
5.11.5.4d	Security analysis review	Y	Y	Y	Y
5.11.5.5a	Security in delivery and acceptance process	Y	Y	Y	Y
5.11.5.5b	Security at acceptance review	Y	Y	Y	Y
5.11.5.5c	Software security analysis and risk treatment measures status	Y	Y	Y	Y
5.11.5.6a	Security in the operations and maintenance process	Y	Y	Y	Y
5.11.5.6b	Software security analysis review	Y	Y	Y	Y
5.11.5.6c	Software security performance	Y	Y	Y	Y
5.11.5.6d	Software security during maintenance	Y	Y	Y	Y
5.11.5.6e	Software security during migration	Y	Y	Y	Y
5.11.5.6f	Software security during retirement	Y	Y	Y	Y
5.11.5.6g	Data sanitation	Y	Y	Y	Y
5.11.5.6h	Data sanitation compliance	Y	Y	Y	Y
5.11.5.6i	Data sanitation records	Y	Y	Y	Y

Annex S (informative) General Tailoring

S.1 Tailoring of this Standard

The general requirements for selection and tailoring of applicable standards are defined in ECSS-S-ST-00.

It provides indication on the general way to tailor an ECSS standard, in particular the tailoring process and the tailoring templates. The templates are generic and deserve a more concrete description of the so–called programmatic and technical factors for software:

- Technical factors:
  - novelty of the domain of application;
  - complexity of the software and the system;
  - criticality category;
  - size of the software;
  - reusability required of the software being developed;
  - interface to system development projects;
  - degree of use of COTS or existing software;
  - maturity of the COTS and completeness or stability of the user requirements.
- Operational factors:
  - type of application (e.g. platform, payload, and experiment);
  - number of potential users of the software;
  - criticality of the software;
  - expected lifetime of the software;
  - number of sites where the software is used;
  - operation, maintenance, migration and retirement constraints.
- Management factors:
  - amount of time and effort required to develop the software;
  - budget requirements for implementing and operating the software;
  - accepted risk level for the project;
  - type of life cycle;
  - schedule requirements for delivering the software;
  - number of people required to develop, operate and maintain the software;
  - complexity of the organization;
  - experience of the supplier;
  - financial resource.

For each particular project additional factors can be used. The factors can be evaluated through the characterisation of the project, identifying the features that influence the selection or not of each requirement. The following questions can help characterizing the project:

- Who are the customer, the supplier, the user, the maintainer, and the operator? Does the customer intend to delegate some tasks to the supplier?
- Where is the complexity of the project, in the requirements or in the design?
- What level of validation is necessary? Should the product be perfect at delivery, or is some room allowed for the user to participate to the tests, or is it a prototype to be dropped later on (or reused in the next phase)?
- What level of verification is necessary? Is it necessary to verify the requirements, or the code, or the test definition?
- What visibility into the design is wished? Does the project manager want to know everything on the detailed design and unit test, or does he trust the supplier for a part of the life cycle?
- Consequently, what are the necessary reviews to be selected into the project? Is it acceptable to merge some of them (as QR and AR, or SRR and PDR) or to waive others (such as CDR)?
- How much are COTS involved? Is the project an assembly of COTS products where the emphasis is in the COTS acceptance and integration?
- Is the software critical? Is it included into an hardware environment?
- How is the maintenance organized? Is it included fully in the current contract, or is the maintenance limited to the guarantee period?

Then the requirements in clauses 5 of this Standard are reviewed and placed in a table with an indication if they are applicable or not. The tailoring of this Standard can result in a short document including the project characteristics (as a justification for the tailoring) and the tailoring table.

An educated engineering judgment recommends that some requirement be never tailored out, such that the production of a minimum set of software requirements, a PDR to review them, the production of the code, a validation against requirement and an acceptance.

The tailoring of this Standard is a task of the Customer (see ECSS-P-00 section 5.3). When preparing the Invitation to Tender, the Customer proposes a tailored version of the standard as an indication of the level of software engineering to be applied for the project. However, some tailoring factors (such as criticality, detailed design complexity) may only be known after the grant of the contract. The Supplier is also part of the tailoring process and the resulting document is baselined in the RB (at SRR).

S.2 List of conditional requirements to be customized

This clause indicates the requirements of this Standard that are applicable in some cases, which are explicitly mentioned in the requirements.

5.2.2.3a in case the software includes HMI

5.3.2.4b. when automatically and manually generated code coexist for the interface between them

5.3.4.2b. in case software requirements are baselined before the start of the architectural design for the SWRR

5.3.4.3b. in case software detailed design is baselined before the start of the coding for the DDR

5.3.6.1a for flight software

5.3.6.2a for ground software

5.4.2.2a when in flight modification is needed

5.6.2.2a in case the project warrants an independent validation effort

5.6.3.1c. and 5.6.4.1c. if it can be justified that validation by test cannot be performed

5.7.2.2a if training and support specified in the requirements baseline

5.8.3.5d. if it can be justified that the required percentage cannot be achieved by test execution

5.8.3.5e. in case of no traceability between source code and object code

5.10.2.2a if the spacecraft lifetime goes after the expected obsolescence date of the software engineering environment, for long term maintenance

5.10.6.1a if software needs to be migrated

5.10.6.4a and 5.10.7.3a if parallel operations of the old and new environments are conducted

S.3 List of requirements with customer - supplier agreement

This clause indicates the requirements for which an agreement between the customer and the supplier is needed during the project.

The customer can consider addressing them in his statement of work before contract signature.

5.3.8.2a for the technical budgets and margin computation

5.8.3.5b. code coverage value

5.10.3.1e. maintenance procedures

Annex T (normative) Software maintenance plan (SMP) - DRD

T.1 DRD identification

T.1.1 Requirement identification and source document

The software maintenance plan (SMP) is called from the normative provisions summarized in Table T-1.

Table T-1: SMP traceability to ECSS-E-ST-40 and ECSS-Q-ST-80 clauses

ECSS Standard	Clauses	DRD section
ECSS-E-ST-40	5.10.2.1a	All
	5.10.2.1b	<7.1>, <7.4>, <9.2>, <10>, <13>
	5.10.2.1c	<13>
	5.10.2.1d	<14>, <15>
	5.10.2.2a	All

T.1.2 Purpose and objective

The software maintenance plan is a constituent of the management file (MF). Its purpose is to provide the definition of organizational aspects and management approach to the implementation of the maintenance tasks.

The objective of the software maintenance plan is to describe the approach to the implementation of the maintenance process for a software product. To this purpose, the plan describes the policies and responsibilities of the program/project team as it plans for software maintenance. Policies and responsibilities are usually spelled out in an organisation policy/directives manual and can be referenced here. It is highly recommended, however, that such doctrine be summarised in the plan.

T.2 Expected response

T.2.1 Scope and content

<1> Introduction

ECSS-E-ST-40_0860800

a. The SMP shall contain a description of the purpose, objective, content and the reason prompting its preparation.

<2> Applicable and reference documents

ECSS-E-ST-40_0860801

a. The SMP shall list the applicable and reference documents to support the generation of the document.

<3> Terms, definitions and abbreviated terms

ECSS-E-ST-40_0860802

a. The SMP shall include any additional terms, definition or abbreviated terms used not already defined in any applicable or reference documentation.

<4> Scope and Purpose

ECSS-E-ST-40_0860803

a. The SMP shall describe the processes and procedures necessary to provide the corrective and preventive maintenance of a software product (i.e. corrections, enhancements, improvements).

ECSS-E-ST-40_0860804

b. The SMP shall define the boundaries of the maintenance process, its starting point (receipt of the request) and the end actions (delivery and sign-off).

ECSS-E-ST-40_0860805

c. The SMP shall address the difference between maintenance and development.

<5> Application of the Plan

ECSS-E-ST-40_0860806

a. The SMP shall describe the overall flow of work including descriptions of each process step and their interfaces, and the data flow between processes.

ECSS-E-ST-40_0860807

b. The SMP shall ensure that each step in the process is controlled and measured, and that expected levels of performance are defined.

<6> General Requirements

<6.1> System

ECSS-E-ST-40_0860808

a. The SMP shall describe the mission of the system including mission need and employment, identification of interoperability requirements and system functions description.

ECSS-E-ST-40_0860809

b. The SMP shall describe the system architecture, components and interfaces, hardware and software.

<6.2> Status

ECSS-E-ST-40_0860810

a. The SMP shall identify the initial status of the system and the complete identification of the system with formal and common names, nomenclature, identification number and system abbreviations.

<6.3> Support

ECSS-E-ST-40_0860811

a. The SMP shall describe why support is needed.

NOTE During the projected life of period of the system, corrections and enhancements will be required. Corrective maintenance accommodates latent defects as reported by users. Enhancements or improvements are submitted in order to improve performance and provide additional functionality for the users. As a result, maintenance support is required.

<6.4> Maintainer organisation

ECSS-E-ST-40_0860812

a. The SMP shall identify the maintainer, including the description of the maintainer team, roles and responsibilities.

NOTE This could be the separate Software Maintenance Staff or the Software Development Staff if there is no transition to a separate maintenance organisation.

<6.5> Contracts

ECSS-E-ST-40_0860813

a. The SMP shall describe any contractual protocols between next-higher-level Contractor and Contractor.

<7> Maintenance Concept

<7.1> Concept

ECSS-E-ST-40_0860814

a. The SMP shall describe the maintenance concepts, including:

1. the scope of software maintenance;
2. the tailoring of the post- delivery process;
3. the designation of who will provide maintenance;
4. an estimate of life-cycle costs;
5. the activities of post-delivery software maintenance.

NOTE 1 The contractor develops it early in the development effort with help from the maintainer. Defining the scope of maintenance helps the contractor determine exactly how much support the maintainer will give to the next-higher-level Contractor. Scope relates to how responsive the maintainer will be to the users.

NOTE 2 Different organisations often perform different activities in the post-delivery process. An early attempt is made to identify these organisations and to document them in the maintenance concept. In many cases, a separate maintenance organisation performs the maintenance functions.

NOTE 3 Responsiveness to the user community is the primary consideration in determining the scope of software maintenance. The scope of software maintenance is tailored to satisfy operational response requirements. Scope relates to how responsive the Maintainer will be to proposed changes. For example, a full scope, software maintenance concept suggests that the Maintainer will provide full support for the entire deployment phase. This includes responding to all approved software change categories (i.e.: corrections and enhancements) within a reasonable period. Software maintenance concepts that limit the scope of software maintenance are referred to as limited scope concepts. Limited scope concepts limit the support period, the support level or both.

<7.2> Level of support

ECSS-E-ST-40_0860815

a. The SMP shall describe the level of support for the system, ensuring that the following requirements are met:

NOTE 1 All corrective and enhancements approved by the Software Change Control Board are included in releases.

NOTE 2 Tracking of all change requests is done.

NOTE 3 A Help Desk is maintained and technical support is provided as needed.

<7.3> Support period

ECSS-E-ST-40_0860816

a. The SMP shall describe the support period from pre-delivery to post-delivery support, on an on-call basis, to review in particular requirements, and plans.

<7.4> Tailoring the maintenance process

ECSS-E-ST-40_0860817

a. The SMP shall describe the tailoring of the maintenance process by referring to the maintainer's software maintenance process manual.

<8> Maintenance Activities

ECSS-E-ST-40_0860818

a. The SMP shall specify the specific maintenance activities.

NOTE General software engineering activities are performed during pre-delivery and post-delivery. The role of the user is defined as well as any interfaces with other organisations.

<9> Resources, methods and standards

<9.1> Resources

ECSS-E-ST-40_0860819

a. The SMP shall analyse the hardware and software most appropriate to support the organisation's needs, including:

1. The definition of the development, maintenance, and target platforms;
2. The description of the differences between the environments;
3. Identification and provision of the tools sets that enhance productivity, of the way the tools are accessible, and the sufficient level of training to users;
4. Description of planning of design, implementation and testing (including associated documentation).

<9.2> Methods and standards

ECSS-E-ST-40_0860820

a. The SMP shall describe the methods and standards to be used for maintenance, in accordance with the ones applied during development process and with the applicable standards.

<10> Maintenance Process

<10.1> Description

ECSS-E-ST-40_0860821

a. The SMP shall describe how modification request are evaluated to determine its classification and handling priority and assignment for implementation as a block of modifications that will be released to the user.

ECSS-E-ST-40_0860822

b. The SMP shall describe the Software Configuration Control Board (i.e. participants, roles, activities).

ECSS-E-ST-40_0860823

c. The SMP shall describe the Maintenance process phases, including:

1. Analysis phase;
2. Design phase;
3. Implementation phase;
4. Acceptance test phase;
5. Delivery phase.

<10.2> Analysis phase

ECSS-E-ST-40_0860824

a. The SMP shall describe the analysis phase including the activities to:

1. Determine if additional problem analysis/identification is required;
2. Record acceptance or rejection of the proposed modification(s);
3. Develop an agreed-upon project plan;
4. Evaluate any software or hardware constraints that may result from the changes and that need consideration during the design phase;
5. Document any project or software risks resulting from the analysis to be considered for subsequent phases of the change life cycle.

NOTE The analysis phase for a modification request can generate several system-level functional, performance, usability, reliability, and maintainability requirements. Each of these may be further decomposed into several software, database, interface, documentation, and hardware requirements.

<10.3> Design phase

ECSS-E-ST-40_0860825

a. The SMP shall describe the design phase.

NOTE 1: Actual implementation begins during this phase, while keeping in mind the continued feasibility of the proposed change. For example, it is possible that the engineering staff does not fully understand the impact and magnitude of a change until the design is complete, or the design of a specific change may be too complex to implement.

NOTE 2: The phase can describe portions of current design specification, software development files, and entries in software engineering case tool database. Other items that can described during this phase include a revised analysis, revised statements of requirements, a revised list of elements affected, a revised plan for implementation, and a revised risk analysis.

<10.4> Implementation phase

ECSS-E-ST-40_0860826

a. The SMP shall describe the implementation phase.

NOTE The primary inputs to this phase are the result of the design phase. Other inputs to describe for successful control of this phase include the following:

- Approved and controlled requirements and design documentation;
- Any design metrics/measurement that may be applicable to the implementation phase;
- A detailed implementation schedule, nothing how many code reviews will take place and at what level;
- A set of responses to the defined risks from the previous phase that are applicable to the testing phase.

Risk analysis and review is performed periodically during this phase rather than at its end, as in the design and analysis phases. This is recommended because a high percentage of design, cost, and performance problems and risks are exposed while modifying the system.

<10.5> Acceptance phase

ECSS-E-ST-40_0860827

a. The SMP shall describe the acceptance test phase, ensuring that the products of the modification are satisfactory to the next-higher-level Contractor, and including the description of:

1. the software system and the documentation necessary to support it;
2. the tests to validate that faults are not introduced as a result of changes;
3. the use of simulations where it is not possible to have the completely integrated system in the test facility.

NOTE The test organisation is responsible for reporting the status of the criteria that had been established in the test plan for satisfactory completion of acceptance testing.

<10.6> Delivery phase

ECSS-E-ST-40_0860828

a. The SMP shall describe the delivery phase including how replacing the existing system with the new version, how to reduce the risk associated with installation of the new version of a software system.

ECSS-E-ST-40_0860829

b. The SMP shall plan the necessary documentation, the user training, and the backup of the existing system version as well as the new version.

NOTE The backup consists of source code, requirement documentation, design documentation, test documentation and the support environment.

<11> Training Requirements

ECSS-E-ST-40_0860830

a. The SMP shall identify the training activities necessary to meet the needs of the maintenance activities.

<12> Software Product Assurance Activities

ECSS-E-ST-40_0860831

a. The SMP shall describe the software product assurance activities for maintenance if not covered by SPAP, including NCRs handling.

ECSS-E-ST-40_0860832

b. The SMP shall describe how qualification status w.r.t. the applicable Standards requirement is maintained and provided.

<13> Software Configuration Management

ECSS-E-ST-40_0860833

a. The SMP shall describe the software configuration management activities for maintenance, including SPRs/SMRs handling, NCR's, CRs and RFWs handling.

<14> Records and Reports

ECSS-E-ST-40_0860834

a. The SMP shall describe the recording, tracking and implementing maintenance including the management and description of the various forms.

NOTE The maintainer documents the problem/modification request, the analysis result and implementation options through the problem analysis report.

ECSS-E-ST-40_0860835

b. The SMP shall describe the rules for submission of maintenance reports.

ECSS-E-ST-40_0860836

c. The SMP shall describe the maintenance records content, including as a minimum the following information:

1. list of request for assistance or problem reports that have been received and the current status of each;
2. Organisation responsible for responding to requests for assistance or implementing the appropriate corrective actions;
3. priorities that have been assigned to the corrective actions;
4. result of corrective and preventive actions;
5. statistical data on failure occurrences and maintenance activities.

NOTE The record of maintenance activities can be utilised for evaluation and enhancement of the software product and for improvement of the quality system itself.

<15> Sample Request Form

ECSS-E-ST-40_0860837

a. The SMP shall describe all template modules necessary to compile the reports, taking into account longer lifetime, evolution or change of the development environment and transfer of the documentation and configuration information to the next project.

Annex U (informative) Software code verification

U.1 Overview

The general requirements for the verification of software code are defined in clause 5.8.3.5. This annex identifies additional recommended checks.

Most of the checks can be covered by a static analysis of the software code. Some checks may need a dynamic analysis, i.e. requiring the execution of the code. All checks aim at ensuring the quality of the software code and at improving the security of the systems that use it.

Some of the checks may be linked to a particular language and only apply to code written in that language.

U.2 Software code verification

The following checks are expected to be performed on the software code:

1. the code implements numerical protection mechanisms (e.g. against overflow and underflow, division by zero);
2. the code does not perform out of bounds accesses - e.g. underrun or overrun of buffers, arrays or strings;
3. the code does not include any infinite loop other than the main loop of the software and the main loops of cyclic tasks;
4. the code appropriately uses arithmetical and logical operators (e.g. arithmetical OR vs. logical OR);
5. implicit type conversions do not lead to arithmetical errors;
6. the lifetime of variables is consistent with their use;
7. the code makes proper use of static/global functions/variables to enforce the correct level of visibility;
8. the code makes proper use of volatile variables for all variables that can be modified asynchronously (e.g. hardware access, memory-mapped I/O);
9. the code does not perform invalid memory accesses (e.g. NULL dereferences);
10. the code does not access uninitialized variables;
11. the code does not perform unused assignments, unless this is done to trigger HW side-effects;
12. there are no memory leaks;
13. pointer arithmetic is justified and types of operands are consistent;
14. the code does not lead to race conditions.

Bibliography

ECSS-S-ST-00	ECSS system — Description, implementation and general requirements
ECSS-E-ST-10	Space engineering — System engineering
ECSS-E-ST-10-06	Space engineering — System engineering - Technical specification
ECSS-E-ST-70	Space engineering — Grounds systems and operations
ECSS-E-ST-70-01	Space engineering — On board control procedures
ECSS-M-ST-80	Space project management — Risk management
ECSS-M-ST-10	Space project management — Project planning and implementation
ECSS-Q-ST-20	Space product assurance — Quality assurance
ISO 9000:2000	Quality management systems — Fundamentals and vocabulary
ISO 9126-1:2001	Software engineering — Product quality — Part 1: Quality model
ISO/IEC 2382-20:1990	Information technology — Vocabulary — Part 20: System development
ISO/IEC 12207:1995	Information technology — Software life cycle processes
ISO/IEC 27000:2018	Information technology — Security techniques — Information security management systems — Overview and vocabulary
IEEE 610.12-1990	Standard glossary of software engineering terminology
RTCA/DO-178B/ED-12B	Software considerations in airborne systems and equipment certification
RTCA/DO-278/ED-109	Guidelines for communication, navigation, surveillance and air traffic management (CNS/ATM) systems software integrity assurance.