ECSS-Q-ST-80C Rev.1
15 February 2017

Space product assurance

Software product assurance

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

Foreword

This Standard is one of the series of ECSS Standards intended to be applied together for the management, engineering and product assurance in space projects and applications. 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-Q-ST-80C 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 Division
ESTEC, P.O. Box 299,
2200 AG Noordwijk
The Netherlands
Copyright: 2017 © by the European Space Agency for the members of ECSS

Change log

ECSS-Q-80A 19 April 1996	First issue
ECSS-Q-80B 10 October 2003	Second issue
ECSS-Q-ST-80C 6 March 2009	Third issue
ECSS-Q-ST-80C Rev.1 15 February 2017	Third issue, Revision 1 Changes with respect to the previous version are identified with revision tracking. The main changes are: • Implementation of Change Requests to ECSS-Q-ST-80C • Implementation of outcomes of Task Force on Software Criticality Classification • Nomenclature added as clause 3.4 **Added requirements:** 5.2.6.1c; 6.2.2.10a. **Modified requirements:** 6.2.2.1a; 6.2.2.2a; 6.2.2.3b NOTE; 6.2.29a; 6.2.2.10a; 6.2.3.2a NOTE; 7.3.5a (formatting corrected); 7.4.1a; Table B-1; B.2.1<5.9>a. and b. (obsolete number in front of requirement text removed), Table C-1; Annex D (several updates); Tables in Annex F (clause references updated). And interleaved NOTES moved at the end of requirement without changing the requirement itself: 5.5.3a; 6.2.3.4a; 6.2.7.4a; 6.2.8.1a; 6.3.5.1a; 7.1.4a. **Deleted requirements:** 6.2.3.1a-b. **Editorial corrections:** Scope, Note in definition 3.2.7, D.1.

Table of contents

Change log **3**

1 Scope **9**

2 Normative references **10**

3 Terms, definitions and abbreviated terms **11**

3.1 Terms for other standards 11

3.2 Terms specific to the present standard 11

3.3 Abbreviated terms 16

3.4 Nomenclature 18

4 Space system software product assurance principles **20**

4.1 Introduction 20

4.2 Organization of this Standard 21

4.3 Tailoring of this Standard 23

5 Software product assurance programme implementation **24**

5.1 Organization and responsibility 24

5.1.1 Organization 24

5.1.2 Responsibility and authority 24

5.1.3 Resources 25

5.1.4 Software product assurance manager/engineer 25

5.1.5 Training 25

5.2 Software product assurance programme management 26

5.2.1 Software product assurance planning and control 26

5.2.2 Software product assurance reporting 27

5.2.3 Audits 28

5.2.4 Alerts 28

5.2.5 Software problems 28

5.2.6 Nonconformances 29

5.2.7 Quality requirements and quality models 29

5.3 Risk management and critical item control 30

5.3.1 Risk management 30
5.3.2 Critical item control 30
5.4 Supplier selection and control 30
5.4.1 Supplier selection 30
5.4.2 Supplier requirements 31
5.4.3 Supplier monitoring 31
5.4.4 Criticality classification 32
5.5 Procurement 32
5.5.1 Procurement documents 32
5.5.2 Review of procured software component list 32
5.5.3 Procurement details 32
5.5.4 Identification 32
5.5.5 Inspection 33
5.5.6 Exportability 33
5.6 Tools and supporting environment 33
5.6.1 Methods and tools 33
5.6.2 Development environment selection 34
5.7 Assessment and improvement process 34
5.7.1 Process assessment 34
5.7.2 Assessment process 35
5.7.3 Process improvement 36

6 Software process assurance 37

6.1 Software development life cycle 37
6.1.1 Life cycle definition 37
6.1.2 Process quality objectives 37
6.1.3 Life cycle definition review 37
6.1.4 Life cycle resources 37
6.1.5 Software validation process schedule 38
6.2 Requirements applicable to all software engineering processes 38
6.2.1 Documentation of processes 38
6.2.2 Software dependability and safety 39
6.2.3 Handling of critical software 41
6.2.4 Software configuration management 43
6.2.5 Process metrics 45
6.2.6 Verification 46
6.2.7 Reuse of existing software 49
6.2.8 Automatic code generation 52

6.3 Requirements applicable to individual software engineering processes or activities....53
6.3.1 Software related system requirements process....53
6.3.2 Software requirements analysis....53
6.3.3 Software architectural design and design of software items....55
6.3.4 Coding....56
6.3.5 Testing and validation....57
6.3.6 Software delivery and acceptance....62
6.3.7 Operations....63
6.3.8 Maintenance....64

7 Software product quality assurance....66
7.1 Product quality objectives and metrication....66
7.1.1 Deriving of requirements....66
7.1.2 Quantitative definition of quality requirements....66
7.1.3 Assurance activities for product quality requirements....66
7.1.4 Product metrics....66
7.1.5 Basic metrics....67
7.1.6 Reporting of metrics....67
7.1.7 Numerical accuracy....67
7.1.8 Analysis of software maturity....68
7.2 Product quality requirements....68
7.2.1 Requirements baseline and technical specification....68
7.2.2 Design and related documentation....69
7.2.3 Test and validation documentation....69
7.3 Software intended for reuse....70
7.3.1 Customer requirements....70
7.3.2 Separate documentation....70
7.3.3 Self-contained information....70
7.3.4 Requirements for intended reuse....70
7.3.5 Configuration management for intended reuse....70
7.3.6 Testing on different platforms....71
7.3.7 Certificate of conformance....71
7.4 Standard ground hardware and services for operational system....71
7.4.1 Hardware procurement....71
7.4.2 Service procurement....71
7.4.3 Constraints....72
7.4.4 Selection....72

7.4.5 Maintenance 72
7.5 Firmware 72
7.5.1 Device programming 72
7.5.2 Marking 73
7.5.3 Calibration 73

Annex A (informative) Software documentation 74

Annex B (normative) Software product assurance plan (SPAP) - DRD 80
B.1 DRD identification 80
B.1.1 Requirement identification and source document 80
B.1.2 Purpose and objective 81
B.2 Expected response 82
B.2.1 Scope and content 82
B.2.2 Special remarks 86

Annex C (normative) Software product assurance milestone report (SPAMR) - DRD 87
C.1 DRD identification 87
C.1.1 Requirement identification and source document 87
C.1.2 Purpose and objective 88
C.2 Expected response 88
C.2.1 Scope and content 88
C.2.2 Special remarks 89

Annex D (normative) Tailoring of this Standard based on software criticality 90
D.1 Software criticality categories 90
D.2 Applicability matrix 92

Annex E (informative) List of requirements with built-in tailoring capability 102

Annex F (informative) Document organization and content at each milestone 103
F.1 Introduction 103
F.2 ECSS-Q-ST-80 Expected Output at SRR 103
F.3 ECSS-Q-ST-80 Expected Output at PDR 105
F.4 ECSS-Q-ST-80 Expected Output at CDR 110
F.5 ECSS-Q-ST-80 Expected Output at QR 112
F.6 ECSS-Q-ST-80 Expected Output at AR 113

F.7 ECSS-Q-ST-80 Expected Output not associated with any specific milestone review 115

Bibliography 117

Figures

Figure 4-1: Software related processes in ECSS Standards 21

Figure 4-2: Structure of this Standard 22

Figure A-1 : Overview of software documents 74

Tables

Table A-1 : ECSS-E-ST-40 and ECSS-Q-ST-80 Document requirements list (DRL) 75

Table B-1 : SPAP traceability to ECSS-E-ST-40 and ECSS-Q-ST-80 clauses 80

Table C-1 : SPAMR traceability to ECSS-Q-ST-80 clauses 87

Table D-1 : Software criticality categories 91

Table D-2 : Applicability matrix based on software criticality 92

1
Scope

This Standard defines a set of software product assurance requirements to be used for the development and maintenance of software for space systems. Space systems include manned and unmanned spacecraft, launchers, payloads, experiments and their associated ground equipment and facilities. Software includes the software component of firmware.

This Standard also applies to the development or reuse of non-deliverable software which affects the quality of the deliverable product or service provided by a space system, if the service is implemented by software.

ECSS-Q-ST-80 interfaces with space engineering and management, which are addressed in the Engineering (-E) and Management (-M) branches of the ECSS System, and explains how they relate to the software product assurance processes.

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

Tailoring of this Standard to a specific business agreement or project, when software product assurance requirements are prepared, is also addressed in clause 4.3.

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-40	Space engineering — Software general requirements
ECSS-Q-ST-10	Space product assurance – Product assurance management
ECSS-Q-ST-10-04	Space product assurance – Critical-item control
ECSS-Q-ST-10-09	Space product assurance – Nonconformance control system
ECSS-Q-ST-20	Space product assurance – Quality assurance
ECSS-Q-ST-30	Space product assurance – Dependability
ECSS-Q-ST-40	Space product assurance – Safety
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-M-ST-80	Space project management – Risk management
ISO/IEC 15504 Part 2:2003	Software engineering - Process assessment – Part 2: Performing an assessment - First Edition

3
Terms, definitions and abbreviated terms

3.1 Terms for other standards

For the purpose of this Standard, the terms and definitions from ECSS-ST-00-01 apply in particular for the term:

acceptance test

software product

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

3.2 Terms specific to the present standard

3.2.1 automatic code generation

generation of source code with a tool from a model

3.2.2 code coverage

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

3.2.3 competent assessor

person who has demonstrated the necessary skills, competencies and experience to lead a process assessment in conformance with ISO/IEC 15504

NOTE Adapted from ISO/IEC 15504:1998, Part 9.

3.2.4 condition

boolean expression not containing boolean operators

3.2.5 configurable code

code (source code or executable code) that can be tailored by setting values of parameters

NOTE This definition covers in particular classes of configurable code obtained by the following configuration means:

- configuration based on the use of a compilation directive;

- configuration based on the use of a link directive;
- configuration performed through a parameter defined in a configuration file;
- configuration performed through data defined in a database with impact on the actually executable parts of the software (e.g. parameters defining branch structures that result in the non-execution of existing parts of the code).

3.2.6 COTS, OTS, MOTS software

for the purpose of this Standard, commercial-off-the-shelf, 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 Annex D.1 – Software criticality categories.

3.2.8 deactivated code

code that, although incorporated through correct design and coding, is intended to execute in certain software product configurations only, or in none of them

[adapted from RTCA/DO-178B]

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, OTS 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

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 mission products

products and services delivered by the space system

NOTE For example: Communications services, science data.

3.2.18 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

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

3.2.19 operational

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

NOTE It is not related to the spacecraft operation.

3.2.20 portability (a quality characteristic)

capability of software to be transferred from one environment to another

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 singular input

input corresponding to a singularity of the function

3.2.27 software

see “software product” in ECSS-S-ST-00-01

3.2.28 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.

3.2.29 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.

3.2.30 software item

see "software product" in ECSS-S-ST-00-01

3.2.31 software observability

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

3.2.32 software problem

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

[CMU/SEI-92-TR-022]

3.2.33 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

3.2.34 software unit

separately compilable piece of source code

NOTE In this Standard no distinction is made between a software unit and a database; both are covered by the same requirements.

3.2.35 statement coverage

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

3.2.36 stress test

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

3.2.37 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

3.2.38 test design

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

3.2.39 test procedure

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

3.2.40 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)

3.2.41 unit test

test of individual software unit

3.2.42 unreachable code

code that cannot be executed due to design or coding error

3.2.43 usability (a quality characteristic)

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

3.2.44 validation

<software> process to confirm that the requirements baseline functions and performances are correctly and completely implemented in the final product

3.2.45 verification

<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

3.2.46 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 1028-1997]

3.3 Abbreviated terms

For the purpose of this Standard and of ECSS-E-ST-40, 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.
CDR	critical design review NOTE The term SW-CDR can be used for clarity to denote CDRs that solely involve software products.
CMMI	capability maturity model integration
COTS	commercial-off-the-shelf
CPU	central processing unit
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
GS	ground segment
HMI	human machine interface
HSIA	hardware-software interaction analysis
HW	hardware
ICD	interface control document
INTRSA	international registration scheme for assessors
IRD	interface requirements document
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

Abbreviation	Meaning
	to denote PDRs that solely involve software products.
PRR	preliminary requirement review
QR	qualification review NOTE The term SW-QR can be used for clarity to denote QRs that solely involve software products.
RB	requirements baseline
SCAMPI	standard CMMI appraisal method for process improvement
SDE	software development environment
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.
SW	software
SWE	software engineering
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 product assurance principles

4.1 Introduction

The objectives of software product assurance are to provide adequate confidence to the customer and to the supplier that the developed or procured/reused software satisfies its requirements throughout the system lifetime. In particular, that the software is developed to perform properly and safely in its operational environment, meeting the quality objectives agreed for the project.

This Standard contributes to these objectives by defining the software product assurance requirements to be met in a particular space project. These requirements deal with quality management and framework, life cycle activities and process definition and quality characteristics of products.

One of the fundamental principles of this Standard is the customer-supplier relationship, assumed for all software developments. The organizational aspects of this are defined in ECSS-M-ST-10. The customer is, in the general case, the procurer of two strongly associated products: the hardware and the software components of a system, subsystem, set, equipment or assembly. The concept of the customer-supplier relationship is applied recursively, i.e. the customer can himself be a supplier to a higher level in the space system hierarchy.

The requirements of this Standard are applicable to the supplier, unless otherwise explicitly stated.

The supplier demonstrates compliance with the software product assurance requirements and provides the specified evidence of compliance.

To this end, the supplier specifies the software product assurance requirements for his/her suppliers, taking into account their responsibilities and the specific nature of their deliverables.

This Standard complements ECSS-E-ST-40 “Space engineering — Software general requirements”, with product assurance aspects, integrated in the space system software engineering processes as defined in ECSS-E-ST-40. Together the two standards specify all processes for space software development.

Figure 4-1 schematically presents the different Software processes addressed by the set of the ECSS standards.

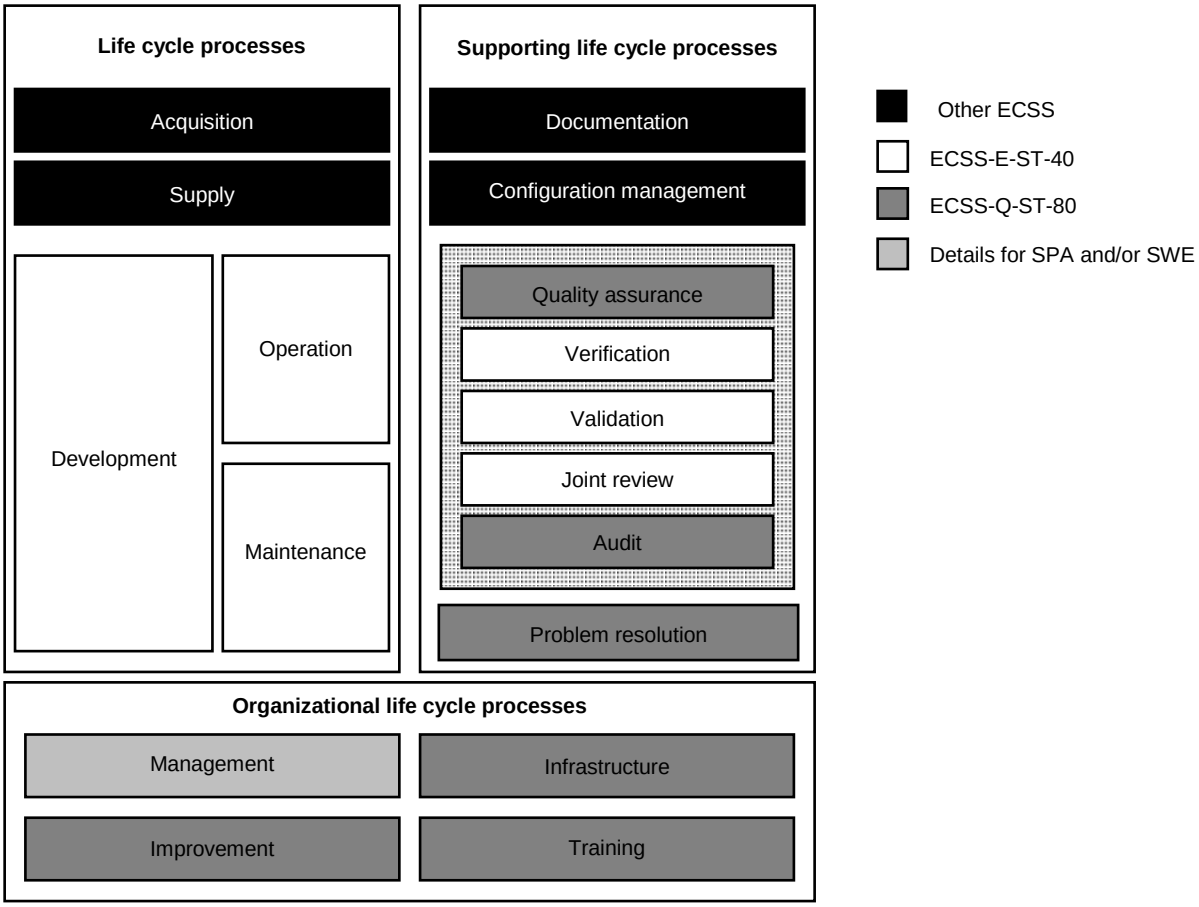

Figure 4-1: Software related processes in ECSS Standards

4.2 Organization of this Standard

This Standard is organized into three main parts:

- Software product assurance programme implementation
- Software process assurance
- Software product quality assurance.

The software documentation collecting the expected output of the ECSS-E-ST-40 and ECSS-Q-ST-80 requirements is summarized in Annex A.

Annex B and Annex C specify the DRDs (document requirements definitions) of the software product assurance documents (SPAP and SPAMR). The DRDs of other software engineering and management documents are included in ECSS-E-ST-40 and ECSS-M-ST-40.

In the preparation of this Standard the ISO/IEC 12207 standard has been used extensively, providing a common internationally recognized framework for the terminology and software life cycle processes description.

The organization of this Standard is reflected in detail in Figure 4-2.

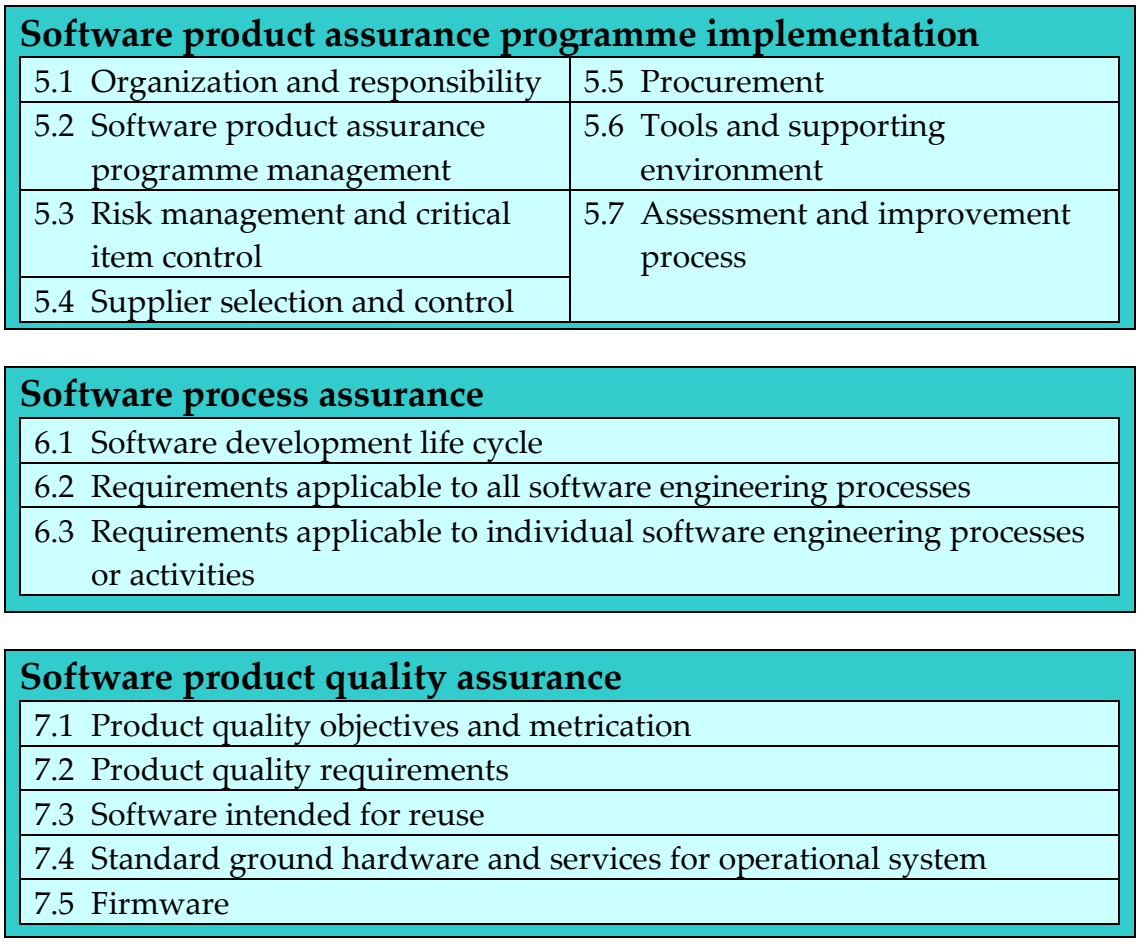

Figure 4-2: Structure of this Standard

Each requirement of this Standard is identified by a hierarchical number, plus a letter if necessary (e.g. 5.3.1.5, bullet a). For each requirement, the associated output is given in the "Expected Output" section. When several outputs are expected, they are identified by a letter (e.g. "a", "b", etc.). With each output, the destination file of the output is indicated in brackets, together with the corresponding document DRD (after a comma) and review(s) (after a semicolon). For example: "*[PAF, SPAP; SRR]*" denotes an output contained in the Software Product Assurance Plan, part of the Product Assurance File, and required at SRR. When no DRD is defined for an Expected Output, and/or the Expected Output is not to be provided at any specific milestone review, then the corresponding sections of that Expected Output are replaced by dashes (e.g. "*[PAF, -; -]*").

This standards details for the Software Product Assurance aspects some of the general requirements already addressed by the ECSS Management, Product Assurance and Quality Assurance standards.

4.3 Tailoring of this Standard

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

There are several drivers for tailoring, such as 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 proofs demanded by the criticality of the software. Annex D contains a tailoring of this Standard based on software criticality.

Tailoring for software development constraints takes into account 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 devices, or application-specific integrated circuits) are also taken into account (see Annex S of ECSS-E-ST-40). 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 product as explained in clause 7 of this Standard based on the quality objectives for the product specified by the customer.

5
Software product assurance programme implementation

5.1 Organization and responsibility

5.1.1 Organization

a. The supplier shall ensure that an organizational structure is defined for software development, and that individuals have defined tasks and responsibilities.

5.1.2 Responsibility and authority

5.1.2.1

a. The responsibility, the authority and the interrelation of personnel who manage, perform and verify work affecting software quality shall be defined and documented.

EXPECTED OUTPUT: *Software product assurance plan [PAF, SPAP; SRR].*

5.1.2.2

a. The responsibilities and the interfaces of each organisation, either external or internal, involved in a project shall be defined and documented.

EXPECTED OUTPUT: *Software product assurance plan [PAF, SPAP; SRR].*

5.1.2.3

a. The delegation of software product assurance tasks by a supplier to a lower level supplier shall be done in a documented and controlled way, with the supplier retaining the responsibility towards the customer.

EXPECTED OUTPUT: *Software product assurance plan [PAF, SPAP; SRR].*

5.1.3 Resources

5.1.3.1

a. The supplier shall provide adequate resources to perform the required software product assurance tasks.

EXPECTED OUTPUT: Software product assurance plan [PAF, SPAP; SRR].

5.1.3.2

a. Reviews and audits of processes and of products shall be carried out by personnel not directly involved in the work being performed.

5.1.4 Software product assurance manager/engineer

5.1.4.1

a. The supplier shall identify the personnel responsible for software product assurance for the project (SW PA manager/engineer).

EXPECTED OUTPUT: Software product assurance plan [PAF, SPAP; SRR].

5.1.4.2

a. The software product assurance manager/engineer shall

1. report to the project manager (through the project product assurance manager, if any);
2. have organisational authority and independence to propose and maintain a software product assurance programme in accordance with the project software product assurance requirements;
3. have unimpeded access to higher management as necessary to fulfil his/her duties.

5.1.5 Training

5.1.5.1

a. The supplier shall review the project requirements to establish and make timely provision for acquiring or developing the resources and skills for the management and technical staff.

EXPECTED OUTPUT: Training plan [MGT, -; SRR].

5.1.5.2

a. The supplier shall maintain training records.

EXPECTED OUTPUT: Records of training and experience [PAF, -; -].

5.1.5.3

a. The supplier shall ensure that the right composition and categories of appropriately trained personnel are available for the planned activities and tasks in a timely manner.

5.1.5.4

a. The supplier shall determine the training subjects based on the specific tools, techniques, methodologies and computer resources to be used in the development and management of the software product.

NOTE Personnel can undergo training to acquire skills and knowledge relevant to the specific field with which the software is to deal.

5.2 Software product assurance programme management

5.2.1 Software product assurance planning and control

5.2.1.1

a. The supplier shall develop a software product assurance plan in response to the software product assurance requirements in conformance with DRD in annex B.

b. The software product assurance plan shall be either a standalone document or a section of the supplier overall product assurance plan.

EXPECTED OUTPUT: Software product assurance plan [PAF, SPAP; SRR, PDR].

5.2.1.2

a. Any internal manuals, standards or procedures referred to by the software product assurance plan shall become an integral part of the supplier's software product assurance programme.

5.2.1.3

a. The software product assurance plan shall be revisited and updated as needed at each milestone to ensure that the activities to be undertaken in the following phase are fully defined.

EXPECTED OUTPUT: Software product assurance plan [PAF, SPAP; CDR, QR, AR, ORR].

5.2.1.4

a. Before acceptance review, the supplier shall either supplement the software product assurance plan to specify the quality measures related

to the operations and maintenance processes, or issue a specific software product assurance plan.

EXPECTED OUTPUT: *Software product assurance plan [PAF, SPAP; AR].*

5.2.1.5

a. The supplier shall provide with the software product assurance plan a compliance matrix documenting conformance with the individual software product assurance requirements applicable for the project or business agreement.

EXPECTED OUTPUT: *Software product assurance plan [PAF, SPAP; SRR, PDR].*

b. For each software product assurance requirement, the compliance matrix shall provide a reference to the document where the expected output of that requirement is located.

NOTE For compliance with the required DRDs a general statement of compliance is acceptable.

EXPECTED OUTPUT: *Software product assurance plan [PAF, SPAP; SRR, PDR].*

5.2.2 Software product assurance reporting

5.2.2.1

a. The supplier shall report on a regular basis on the status of the software product assurance programme implementation, if appropriate as part of the overall product assurance reporting of the project.

EXPECTED OUTPUT: *Software product assurance reports [PAF, -; -].*

5.2.2.2

a. The software product assurance report shall include:

1. an assessment of the current quality of the product and processes, based on measured properties, with reference to the metrication as defined in the software product assurance plan;
2. verifications undertaken;
3. problems detected;
4. problems resolved.

EXPECTED OUTPUT: *Software product assurance reports [PAF, -; -].*

5.2.2.3

a. The supplier shall deliver at each milestone review a software product assurance milestone report, covering the software product assurance activities performed during the past project phases.

EXPECTED OUTPUT: *Software product assurance milestone report [PAF, SPAMR; SRR, PDR, CDR, QR, AR, ORR].*

5.2.3 Audits

a. For software audits, ECSS-Q-ST-10 clause 5.2.3 shall apply.

EXPECTED OUTPUT: *Audit plan and schedule [PAF, -; SRR].*

5.2.4 Alerts

a. For software alerts, ECSS-Q-ST-10 clause 5.2.9 shall apply.

EXPECTED OUTPUT: *The following outputs are expected:*

a. Preliminary alert information [PAF, -; -];

b. Alert information [PAF, -; -].

5.2.5 Software problems

5.2.5.1

a. The supplier shall define and implement procedures for the logging, analysis and correction of all software problems encountered during software development.

EXPECTED OUTPUT: *Software problem reporting procedures [PAF, -; PDR].*

5.2.5.2

a. The software problem report shall contain the following information:

1. identification of the software item;
2. description of the problem;
3. recommended solution;
4. final disposition;
5. modifications implemented (e.g. documents, code, and tools);
6. tests re-executed.

EXPECTED OUTPUT: *Software problem reporting procedures [PAF, -; PDR].*

5.2.5.3

a. The procedures for software problems shall define the interface with the nonconformance system (i.e. the circumstances under which a problem qualifies as a nonconformance).

EXPECTED OUTPUT: *Software problem reporting procedures [PAF, -; PDR].*

5.2.5.4

a. The supplier shall ensure the correct application of problem reporting procedures.

5.2.6 Nonconformances

5.2.6.1

a. For software nonconformance handling, ECSS-Q-ST-10-09 shall apply.

EXPECTED OUTPUT: *The following outputs are expected:*

a. NCR SW procedure as part of the Software product assurance plan [PAF, SPAP; SRR];

b. Nonconformance reports [DJF, -; -].

b. When dealing with software nonconformance, the NRB shall include, at least, a representative from the software product assurance and the software engineering organizations.

EXPECTED OUTPUT: *Identification of SW experts in NRB [MGT, -; SRR]*

c. The NRB shall dispose software nonconformances according to the following criteria:

1. use "as-is", when the software is found to be usable without eliminating the nonconformance;
2. fix, when the software product can be made fully in conformance with all specified requirements, by:
   (a) correction of the software,
   (b) addition of software patches, or
   (c) re-design.
3. return to supplier, for procured software products (e.g. COTS).

EXPECTED OUTPUT: *Nonconformance reports [DJF, -; -].*

5.2.6.2

a. The software product assurance plan shall specify the point in the software life cycle from which the nonconformance procedures apply.

EXPECTED OUTPUT: *Software product assurance plan [PAF, SPAP; SRR, PDR].*

5.2.7 Quality requirements and quality models

5.2.7.1

a. Quality models shall be used to specify the software quality requirements.

EXPECTED OUTPUT: *Software product assurance plan [PAF, SPAP; PDR].*

5.2.7.2

a. The following characteristics shall be used to specify the quality model:

1. functionality;

2. reliability;
3. maintainability;
4. reusability;
5. suitability for safety;
6. security;
7. usability;
8. efficiency;
9. portability;
10. software development effectiveness.

NOTE 1 Quality models are the basis for the identification of process metrics (see clause 6.2.5) and product metrics (see clause 7.1.4).

NOTE 2 quality models are also addressed by ISO/IEC 9126 or ECSS-Q-HB-80-04.

EXPECTED OUTPUT: *Software product assurance plan [PAF, SPAP; PDR].*

5.3 Risk management and critical item control

5.3.1 Risk management

a. Risk management for software shall be performed by cross-reference to the project risk policy, as specified in ECSS-M-ST-80.

5.3.2 Critical item control

5.3.2.1

a. For critical item control, ECSS-Q-ST-10-04 shall apply.

5.3.2.2

a. The supplier shall identify the characteristics of the software items that qualify them for inclusion in the Critical Item List.

5.4 Supplier selection and control

5.4.1 Supplier selection

5.4.1.1

a. For supplier selection ECSS-Q-ST-20 clause 5.4.1 shall apply.

EXPECTED OUTPUT: *The following outputs are expected:*

a. Results of pre-award audits and assessments [PAF, -; -];

b. Records of procurement sources [PAF, -; -].

5.4.1.2

a. For the selection of suppliers of existing software, including software contained in OTS equipments and units, the expected output of clauses 6.2.7.2 to 6.2.7.6 shall be made available.

EXPECTED OUTPUT: *Software reuse file [DJF, SRF; -].*

5.4.2 Supplier requirements

5.4.2.1

a. The supplier shall establish software product assurance requirements for the next level suppliers, tailored to their role in the project, including a requirement to produce a software product assurance plan.

EXPECTED OUTPUT: *Software product assurance requirements for suppliers [PAF, -; SRR].*

5.4.2.2

a. The supplier shall provide the software product assurance requirements applicable to the next level suppliers for customer's acceptance.

EXPECTED OUTPUT: *Software product assurance requirements for suppliers [PAF, -; SRR].*

5.4.3 Supplier monitoring

5.4.3.1

a. The supplier shall monitor the next lower level suppliers' conformance to the product assurance requirements.

5.4.3.2

a. The monitoring process shall include the review and approval of the next lower level suppliers' product assurance plans, the continuous verification of processes and products, and the monitoring of the final validation of the product.

5.4.3.3

a. The supplier shall ensure that software development processes are defined and applied by the next lower level suppliers in conformance with the software product assurance requirements for suppliers.

EXPECTED OUTPUT: *Next level suppliers' software product assurance plan [PAF, SPAP; PDR].*

5.4.3.4

a. The supplier shall provide the next lower level suppliers' software product assurance plan for customer's acceptance.

EXPECTED OUTPUT: *Next level suppliers' software product assurance plan [PAF, SPAP; PDR].*

5.4.4 Criticality classification

a. The supplier shall provide the lower level suppliers with the relevant results of the safety and dependability analyses performed at higher and his level (ref. clauses 6.2.2.1 and 6.2.2.2), including:

1. the criticality classification of the software products to be developed;
2. information about the failures that can be caused at higher level by the software products to be developed.

EXPECTED OUTPUT: *Safety and dependability analyses results for lower level suppliers [RB, -; SRR].*

5.5 Procurement

5.5.1 Procurement documents

a. For procurement documents, ECSS-Q-ST-20 clause 5.4.2 shall apply.

5.5.2 Review of procured software component list

a. The choice of procured software shall be described and submitted for customer review.

EXPECTED OUTPUT: *Software development plan [MGT, SDP; SRR, PDR].*

5.5.3 Procurement details

a. For each of the software items the following data shall be provided:

1. ordering criteria
2. receiving inspection criteria;
3. back-up solutions if the product becomes unavailable;
4. contractual arrangements with the supplier for the development, maintenance and upgrades to new releases.

NOTE Examples of ordering criteria are: versions, options and extensions.

EXPECTED OUTPUT: *Procurement data [MGT, -; SRR, PDR].*

5.5.4 Identification

a. All the procured software shall be identified and registered by configuration management.

5.5.5 Inspection

a. The supplier shall subject the procured software to a planned receiving inspection, in accordance with ECSS-Q-ST-20 clause 5.4.4, and the receiving inspection criteria as required by clause 5.5.3.

EXPECTED OUTPUT: *Receiving inspection report [PAF, -; PDR, CDR, QR].*

5.5.6 Exportability

a. Exportability constraints shall be identified.

5.6 Tools and supporting environment

5.6.1 Methods and tools

5.6.1.1

a. Methods and tools to be used for all the activities of the development cycle, (including requirements analysis, software specification, modelling, design, coding, validation, testing, configuration management, verification and product assurance) shall be identified by the supplier and agreed by the customer.

EXPECTED OUTPUT: *Software product assurance plan [PAF, SPAP; SRR, PDR].*

5.6.1.2

a. The choice of development methods and tools shall be justified by demonstrating through testing or documented assessment that:

1. the development team has appropriate experience or training to apply them,
2. the tools and methods are appropriate for the functional and operational characteristics of the product, and
3. the tools are available (in an appropriate hardware environment) throughout the development and maintenance lifetime of the product.

EXPECTED OUTPUT: *Software product assurance milestone report [PAF, SPAMR; SRR, PDR].*

5.6.1.3

a. The correct use of methods and tools shall be verified and reported.

EXPECTED OUTPUT: *Software product assurance reports [PAF, -; -].*

5.6.2 Development environment selection

5.6.2.1

a. The software development environment shall be selected according to the following criteria:

1. availability;
2. compatibility;
3. performance;
4. maintenance;
5. durability and technical consistency with the operational equipment;
6. the assessment of the product with respect to requirements, including the criticality category;
7. the available support documentation;
8. the acceptance and warranty conditions;
9. the conditions of installation, preparation, training and use;
10. the maintenance conditions, including the possibilities of evolutions;
11. copyright and intellectual property rights constraints;
12. dependence on one specific supplier.

EXPECTED OUTPUT: *Software development plan [MGT, SDP; SRR, PDR].*

5.6.2.2

a. The suitability of the software development environment shall be justified.

EXPECTED OUTPUT: *Software development plan [MGT, SDP; SRR, PDR].*

5.6.2.3

a. The availability of the software development environment to developers and other users shall be verified before the start of each development phase.

5.7 Assessment and improvement process

5.7.1 Process assessment

a. The supplier shall monitor and control the effectiveness of the processes used during the development of the software, including the relevant

processes corresponding to the services called from other organizational entities outside the project team.

NOTE The process assessment and improvement performed at organization level can be used to provide evidence of compliance for the project.

EXPECTED OUTPUT: *Software process assessment records: Overall assessments and improvement programme plan [PAF, -; -].*

5.7.2 Assessment process

5.7.2.1

a. The process assessment model and method to be used when performing any software process assessment shall be documented.

EXPECTED OUTPUT: *The following outputs are expected:*

a. Software process assessment record: assessment model [PAF, -; -];

b. Software process assessment record: assessment method [PAF, -; -].

5.7.2.2

a. Assessments performed and process assessment models used shall be in conformance with ISO/IEC 15504 (Part 2).

EXPECTED OUTPUT: *The following outputs are expected:*

a. Software process assessment record: evidence of conformance of the process assessment model [PAF, -; -];

b. Software process assessment record: assessment method [PAF, -; -].

NOTE 1 The model and method documented in ECSS-Q-HB-80-02 are conformant to ISO/IEC 15504 (Part 2).

NOTE 2 Currently the CMMI model is not fully conformant to ISO/IEC 15504, however it can be used, provided that the SCAMPI A method is applied.

5.7.2.3

a. The process assessment model, the method, the assessment scope, the results and the assessors shall be verified as complying with the project requirements.

NOTE 1 Examples of assessment scopes are: organizational unit evaluated, and processes evaluated.

NOTE 2 ECSS-Q-HB-80-02 provides space specific process reference model and their indicators.

EXPECTED OUTPUT: *Software process assessment record: Software process assessment recognition evidence [PAF, -; -].*

5.7.2.4

a. Assessments, carried out in accordance with ECSS-Q-HB-80-02, shall be performed by a competent assessor, whereas the other assessment team members can be either competent assessor or provisional assessor.

NOTE 1 For other assessment schemes conformant to ISO/IEC 15504 (Part 2), assessors certified under INTRSA are competent assessors.

NOTE 2 When using CMMI/SCAMPI A, SEI authorized lead appraisers are competent assessors.

EXPECTED OUTPUT: *Software process assessment record: competent assessor justification [PAF, -; -].*

5.7.3 Process improvement

5.7.3.1

a. The results of the assessment shall be used as feedback to improve as necessary the performed processes, to recommend changes in the direction of the project, and to determine technology advancement needs.

b. The suppliers shall ensure that the results of previous assessments are used in its project activity

EXPECTED OUTPUT: *Software process assessment records: improvement plan [PAF, -; -].*

5.7.3.2

a. The process improvement shall be conducted according to a documented process improvement process.

NOTE 1 For the definition of the process improvement process, see ECSS-Q-HB-80-02.

NOTE 2 For CMMI, the process improvement is described in the OPF (Organizational Process Focus) process area.

EXPECTED OUTPUT: *Software process assessment records: improvement process [PAF, -; -].*

5.7.3.3

a. Evidence of the improvement in performed processes or in project documentation shall be provided.

NOTE See ECSS-Q-HB-80-02.

EXPECTED OUTPUT: *Software process assessment records: evidence of improvements [PAF, -; -].*

6
Software process assurance

6.1 Software development life cycle

6.1.1 Life cycle definition

a. The software development life cycle shall be defined or referenced in the software product assurance plan.

b. The following characteristics of the software life cycle shall be defined:

1. phases;
2. input and output of each phase;
3. status of completion of phase output;
4. milestones;
5. dependencies;
6. responsibilities;
7. role of the customer at each milestone review, in conformance with ECSS-M-ST-10 and ECSS-M-ST-10-01.

EXPECTED OUTPUT: Software product assurance plan [PAF, SPAP; SRR, PDR].

6.1.2 Process quality objectives

a. In the definition of the life cycle and associated milestones and documents, the quality objectives shall be used.

6.1.3 Life cycle definition review

a. The software life cycle shall be reviewed against the contractual software engineering and product assurance requirements.

6.1.4 Life cycle resources

a. The software life cycle shall be reviewed for suitability and for the availability of resources to implement it by all functions involved in its application.

6.1.5 Software validation process schedule

a. A milestone (SW TRR as defined in ECSS-E-ST-40 clause 5.3.5.1) shall be scheduled immediately before the software validation process starts, to check that:

1. the software status is compatible with the commencement of validation activities;
2. the necessary resources, software product assurance plans, test and validation documentation, simulators or other technical means are available and ready for use.

EXPECTED OUTPUT: *Software product assurance plan [PAF, SPAP; SRR, PDR].*

6.2 Requirements applicable to all software engineering processes

6.2.1 Documentation of processes

6.2.1.1

a. The following activities shall be covered either in software-specific plans or in project general plans:

1. development;
2. specification, design and customer documents to be produced;
3. configuration and documentation management;
4. verification, testing and validation activities;
5. maintenance.

EXPECTED OUTPUT: *Software project plans [MGT, MF, DJF].*

6.2.1.2

a. All plans shall be finalized before the start of the related activities.

EXPECTED OUTPUT: *Software project plans [MGT, MF, DJF].*

6.2.1.3

a. All plans shall be updated for each milestone to reflect any changes during development.

EXPECTED OUTPUT: *Software project plans [MGT, MF, DJF].*

6.2.1.4

a. The software product assurance plan shall identify all plans to be produced and used, the relationship between them and the time-scales for their preparation and update.

EXPECTED OUTPUT: *Software product assurance plan [PAF, SPAP; SRR, PDR].*

6.2.1.5

a. Each plan shall be reviewed against the relevant contractual requirements.

6.2.1.6

a. Procedures and project standards shall address all types of software products included in the project.

EXPECTED OUTPUT: *Procedures and standards [PAF, -; PDR].*

6.2.1.7

a. All procedures and project standards shall be finalized before starting the related activities.

EXPECTED OUTPUT: *Procedures and standards [PAF, -; PDR].*

6.2.1.8

a. Each procedure or standard shall be reviewed against the relevant plans and contractual requirements.

6.2.1.9

a. Before any activity is started, each procedure or standard for that activity shall be reviewed by all functions involved in its application, for suitability and for the availability of resources to implement it.

6.2.2 Software dependability and safety

6.2.2.1

a. For the system-level analyses leading to the criticality classification of software products based on the severity of failures consequences, ECSS-Q-ST-40 clause 6.5.6.3, and ECSS-Q-ST-30 clause 5.4, shall apply.

EXPECTED OUTPUT: *Criticality classification of software products [PAF, -; SRR, PDR].*

6.2.2.2

a. The supplier shall perform a software dependability and safety analysis of the software products, using the results of system-level safety and dependability analyses, in order to determine the criticality of the individual software components.

EXPECTED OUTPUT: *Software dependability and safety analysis report [PAF, -; PDR].*

6.2.2.3

a. The supplier shall identify the methods and techniques for the software dependability and safety analysis to be performed at technical specification and design level.

b. Methods and techniques for software dependability and safety analysis shall be agreed between the supplier and customer.

NOTE ECSS-Q-HB-80-03 provides indication on methods and techniques that can be applied such as:

- software failure modes and effects analysis (for the performing of this analysis, see also ECSS-Q-ST-30-02);
- software fault tree analysis;
- software common cause failure analysis.

EXPECTED OUTPUT: *Criticality classification of software components [PAF, -; PDR].*

6.2.2.4

a. Based on the results of the software criticality analysis, the supplier shall apply engineering measures to reduce the number of critical software components and mitigate the risks associated with the critical software (ref. clause 6.2.3).

6.2.2.5

a. The supplier shall report on the status of the implementation and verification of the SW dependability and safety analysis recommendations.

EXPECTED OUTPUT: *Software dependability and safety analysis report [PAF, -; CDR, QR, AR].*

6.2.2.6

a. The supplier shall update the software dependability and safety analysis at each software development milestone, to confirm the criticality category of software components.

EXPECTED OUTPUT: *Software dependability and safety analysis report [PAF, -; CDR, QR, AR].*

6.2.2.7

a. The supplier shall provide the results of the software dependability and safety analysis for integration into the system-level dependability and safety analyses, addressing in particular:

1. additional failure modes identified at software design level;
2. recommendations for system-level activities.

NOTE For example: introduction of hardware inhibits, and modifications of the system architecture.

EXPECTED OUTPUT: *Software dependability and safety analysis report [PAF, -; PDR, CDR].*

6.2.2.8

a. As part of the software requirements analysis activities (ref. clause 6.3.2), the supplier shall contribute to the Hardware-Software Interaction Analysis (HSIA) by identifying, for each hardware failure included in the HSIA, the requirements that specify the software behaviour in the event of that hardware failure.

6.2.2.9

a. During the verification and validation of the software requirements resulting from the Hardware-Software Interaction Analysis, the supplier shall verify that the software reacts correctly to hardware failures, and no undesired software behaviour occurs that may lead to system failures.

6.2.2.10

a. If it cannot be prevented that software components cause failures of higher criticality components, due to failure propagation or use of shared resources, then all the involved components shall be classified at the highest criticality category among them.

NOTE Failures of higher-criticality software components caused by lower-criticality components can be prevented by design measures such as separate hardware platforms, isolation of software processes or prohibition of shared memory (segregation and partitioning).

EXPECTED OUTPUT: *The following outputs are expected:*

a. Software product assurance plan [PAF, SPAP; PDR, CDR];

b. Software dependability and safety analysis report [PAF, -; PDR, CDR, QR, AR].

6.2.3 Handling of critical software

6.2.3.1

a. <<deleted>>

b. <<deleted>>

6.2.3.2

a. The supplier shall define, justify and apply measures to assure the dependability and safety of critical software.

NOTE These measures can include:

- use of software design or methods that have performed successfully in a similar application;
- insertion of features for failure isolation and handling (ref. ECSS-Q-HB-80-03, software failure modes and effects analysis);
- defensive programming techniques, such as input verification and consistency checks;
- use of a "safe subset" of programming language;
- use of formal design language for formal proof;
- 100 % code branch coverage at unit testing level;
- full inspection of source code;
- witnessed or independent testing;
- gathering and analysis of failure statistics;
- removing deactivated code or showing through a combination of analysis and testing that the means by which such code can be inadvertently executed are prevented, isolated, or eliminated.

EXPECTED OUTPUT: *Software product assurance plan [PAF, SPAP; PDR, CDR].*

6.2.3.3

a. The application of the chosen measures to handle the critical software shall be verified.

EXPECTED OUTPUT: *Software product assurance milestone report [PAF, SPAMR; PDR, CDR, QR, AR].*

6.2.3.4

a. Critical software shall be subject to regression testing after:

1. any change of functionality of the underlying platform hardware;
2. any change of the tools that affect directly or indirectly the generation of the executable code.

NOTE 1 In case of minor changes in tools that affect the generation of the executable code, a binary comparison of the executable code generated by the different tools can be used to verify that no modifications are introduced.

NOTE 2 Example for item 1: instruction set of a processor.

EXPECTED OUTPUT: *Software product assurance plan [PAF, SPAP; PDR, CDR].*

6.2.3.5

a. The need for additional verification and validation of critical software shall be analysed after:

   1. any change of functionality or performance of the underlying platform hardware;

   2. any change in the environment in which the software or the platform hardware operate.

   EXPECTED OUTPUT: *Software product assurance plan [PAF, SPAP; PDR, CDR].*

6.2.3.6

a. Identified unreachable code shall be removed and the need for re-verification and re-validation shall be analysed.

6.2.3.7

a. Unit and integration testing shall be (re-)executed on non-instrumented code.

6.2.3.8

a. Validation testing shall be (re-)executed on non-instrumented code.

6.2.4 Software configuration management

6.2.4.1

a. ECSS-M-ST-40 shall be applied for software configuration management, complemented by the following requirements.

6.2.4.2

a. The software configuration management system shall allow any reference version to be re-generated from backups.

   EXPECTED OUTPUT: *Software configuration management plan [MGT, SCMP; SRR, PDR].*

6.2.4.3

a. The software configuration file and the software release document shall be provided with each software delivery.

   EXPECTED OUTPUT: *The following outputs are expected:*

   a. Software configuration file [DDF, SCF; -];

   b. Software release document [DDF, SRelD; -].

6.2.4.4

a. The software configuration file shall be available and up to date for each project milestone.

EXPECTED OUTPUT: *Software configuration file [DDF, SCF; CDR, QR, AR, ORR].*

6.2.4.5

a. Any components of the code generation tool that are customizable by the user shall be put under configuration control.

b. The change control procedures defined for the project shall address the specific aspects of these components.

EXPECTED OUTPUT: *The following outputs are expected:*

a. Software configuration file [DDF, SCF; CDR, QR, AR, ORR];

b. Software configuration management plan [MGT, SCMP; SRR, PDR].

6.2.4.6

a. The supplier shall ensure that all authorized changes are implemented in accordance with the software configuration management plan.

EXPECTED OUTPUT: *Authorized changes - Software configuration file [DDF, SCF; CDR, QR, AR, ORR].*

6.2.4.7

a. The following documents shall be controlled (see ECSS-Q-ST-10 clause 5.2.5):

1. procedural documents describing the quality system to be applied during the software life cycle;
2. planning documents describing the planning and progress of the activities;
3. documents describing a particular software product, including:
   (a) development phase inputs,
   (b) development phase outputs,
   (c) verification and validation plans and results,
   (d) test case specifications, test procedures and test reports,
   (e) traceability matrices,
   (f) documentation for the software and system operators and users, and
   (g) maintenance documentation.

6.2.4.8

a. The supplier shall identify a method and tool to protect the supplied software against corruption.

NOTE For example: source, executable and data.

EXPECTED OUTPUT: *The following outputs are expected:*

a. Software product assurance plan [PAF, SPAP; SRR, PDR];

b. Software configuration file [DDF, SCF; CDR, QR, AR, ORR].

6.2.4.9

a. The supplier shall define a checksum-type key calculation for the delivered operational software.

NOTE For example: executable binary, database.

EXPECTED OUTPUT: *Software product assurance plan [PAF, SPAP; SRR, PDR].*

6.2.4.10

a. The checksum value shall be provided in the software configuration file with each software delivery.

EXPECTED OUTPUT: *Software configuration file [DDF, SCF; -].*

6.2.4.11

a. The media through which the software is delivered to the customer shall be marked by the supplier indicating the following information as a minimum:

1. the software name;
2. the version number;
3. the reference to the software configuration file.

EXPECTED OUTPUT: *The following outputs are expected:*

a. Software product assurance plan [PAF, SPAP; SRR, PDR];

b. Labels [DDF, -; -].

6.2.5 Process metrics

6.2.5.1

a. Metrics shall be used to manage the development and to assess the quality of the development processes.

NOTE Process metrics are based on quality models (see clause 5.2.7).

EXPECTED OUTPUT: *Software product assurance plan [PAF, SPAP; SRR, PDR].*

6.2.5.2

a. Process metrics shall be collected, stored and analysed on a regular basis by applying quality models and procedures.

EXPECTED OUTPUT: Software product assurance plan [PAF, SPAP; SRR, PDR].

6.2.5.3

a. The following basic process metrics shall be used within the supplier's organization:

1. duration: how phases and tasks are being completed versus the planned schedule;
2. effort: how much effort is consumed by the various phases and tasks compared to the plan.

EXPECTED OUTPUT: Internal metrics report.

6.2.5.4

a. Process metrics shall be used within the supplier's organization and reported to the customer, including:

1. number of problems detected during verification;
2. number of problems detected during integration and validation testing and use.

NOTE See also software problem reporting described in clause 5.2.5.

EXPECTED OUTPUT: Software product assurance reports [PAF, -; -].

6.2.5.5

a. Metrics reports shall be included in the software product assurance reports.

EXPECTED OUTPUT: Software product assurance reports [PAF, -; -].

6.2.6 Verification

6.2.6.1

a. Activities for the verification of the quality requirements shall be specified in the definition of the verification plan.

NOTE Verification includes various techniques such as review, inspection, testing, walk-through, cross-reading, desk-checking, model simulation, and many types of analysis such as traceability analysis, formal proof or fault tree analysis.

EXPECTED OUTPUT: Software verification plan [DJF, SVerP; PDR].

6.2.6.2

a. The outputs of each development activity shall be verified for conformance against pre-defined criteria.

b. Only outputs which have been subjected to planned verifications shall be used as inputs for subsequent activities.

EXPECTED OUTPUT: *Software product assurance reports [PAF, -; -]*.

6.2.6.3

a. A summary of the assurance activities concerning the verification process and their findings shall be included in software product assurance reports.

EXPECTED OUTPUT: *Software product assurance reports [PAF, -; -]*.

6.2.6.4

a. The completion of actions related to software problem reports generated during verification shall be verified and recorded.

EXPECTED OUTPUT: *Software problem reports [DJF, -; SRR, PDR, CDR, QR, AR, ORR]*.

6.2.6.5

a. Software containing deactivated code shall be verified specifically to ensure that the deactivated code cannot be activated or that its accidental activation cannot harm the operation of the system.

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

6.2.6.6

a. Software containing configurable code shall be verified specifically to ensure that any unintended configuration cannot be activated at run time or included during code generation.

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

6.2.6.7

a. The supplier shall ensure that:

1. the planned verification activities are adequate to confirm that the products of each phase are conformant to the applicable requirements;

2. the verification activities are performed according to the plan.

EXPECTED OUTPUT: *Software product assurance reports [PAF, -; -]*.

6.2.6.8

a. Reviews and inspections shall be carried out according to defined criteria, and according to the defined level of independence of the reviewer from the author of the reviewed item.

6.2.6.9

a. Each review and inspection shall be based on a written plan or procedure.

NOTE For projects reviews, ECSS-E-ST-40 clause 5.3.3.3, bullet b and Annex P are applicable.

EXPECTED OUTPUT: *Review and inspection plans or procedures [PAF, -; -].*

6.2.6.10

a. The review or inspection plans or procedures shall specify:

1. the reviewed or inspected items;
2. the person in charge;
3. the participants;
4. the means of review or inspection (e.g. tools or check list);
5. the nature of the report.

EXPECTED OUTPUT: *Review and inspection plans or procedures [PAF, -; -].*

6.2.6.11

a. Review and inspection reports shall:

1. refer to the corresponding review/inspection procedure or plan;
2. identify the reviewed item, the author, the reviewer, the review criteria and the findings of the review.

EXPECTED OUTPUT: *Review and inspection reports [PAF, -; -].*

6.2.6.12

a. Traceability matrices (as defined in ECSS-E-ST-40 clause 5.8) shall be verified at each milestone.

EXPECTED OUTPUT: *Software product assurance milestone report [PAF, SPAMR; SRR, PDR, CDR, QR, AR, ORR].*

6.2.6.13

a. Independent software verification shall be performed by a third party.

b. Independent software verification shall be a combination of reviews, inspections, analyses, simulations, testing and auditing.

NOTE This requirement is applicable where the risks associated with the project justify the costs involved. The customer can consider a less rigorous level of independence, e.g. an independent team in the same organization.

EXPECTED OUTPUT: *The following outputs are expected:*

a. ISVV plan [DJF, -; SRR, PDR];

b. ISVV report [DJF, -; PDR, CDR, QR, AR].

6.2.7 Reuse of existing software

6.2.7.1 General

The requirements in 6.2.7 do not apply to tools and software development environment, for which requirements of clause 5.6 apply.

6.2.7.2

a. Analyses of the advantages to be obtained with the selection of existing software (ref. 3.2.11) instead of new development shall be carried out.

EXPECTED OUTPUT: *The following outputs are expected:*

a. Software reuse approach, including approach to delta qualification [PAF, SPAP; SRR, PDR];

b. Software reuse file [DJF, SRF; SRR, PDR].

6.2.7.3

a. The existing software shall be assessed with regards to the applicable functional, performance and quality requirements.

EXPECTED OUTPUT: *The following outputs are expected:*

a. Software reuse approach, including approach to delta qualification [PAF, SPAP; SRR, PDR];

b. Software reuse file [DJF, SRF; SRR, PDR].

6.2.7.4

a. The quality level of the existing software shall be analysed with respect to the project requirements, according to the criticality of the system function implemented, taking into account the following aspects:

1. software requirements documentation;
2. software architectural and detailed design documentation;
3. forward and backward traceability between system requirements, software requirements, design and code;
4. unit tests documentation and coverage;
5. integration tests documentation and coverage;
6. validation documentation and coverage;
7. verification reports;
8. performance;

9. operational performances;
10. residual nonconformances and waivers;
11. user operational documentation;
12. code quality (adherence to coding standards, metrics).

NOTE 1 Examples of performance are memory occupation, CPU load.

NOTE 2 Example of user operation documentation is a user manual.

EXPECTED OUTPUT: *The following outputs are expected:*

a. Software reuse approach, including approach to delta qualification [PAF, SPAP; SRR, PDR];

b. Software reuse file [DJF, SRF; SRR, PDR].

6.2.7.5

a. The results of the reused software analysis shall be recorded in the software reuse file, together with an assessment of the possible level of reuse and a description of the assumptions and the methods applied when estimating the level of reuse.

NOTE Results of the reused software analysis, such as detailed reference to requirement and design documents, test reports and coverage results.

EXPECTED OUTPUT: *The following outputs are expected:*

a. Software reuse approach, including approach to delta qualification [PAF, SPAP; SRR, PDR];

b. Software reuse file [DJF, SRF; SRR, PDR].

6.2.7.6

a. The analysis of the suitability of existing software for reuse shall be complemented by an assessment of the following aspects:

1. the acceptance and warranty conditions;
2. the available support documentation;
3. the conditions of installation, preparation, training and use;
4. the identification and registration by configuration management;
5. maintenance responsibility and conditions, including the possibilities of changes;
6. the durability and validity of methods and tools used in the initial development, that are envisaged to be used again;
7. the copyright and intellectual property rights constraints (modification rights);
8. the licensing conditions;
9. exportability constraints.

EXPECTED OUTPUT: *Software reuse file [DJF, SRF; SRR, PDR].*

6.2.7.7

a. Corrective actions shall be identified, documented in the reuse file and applied to the reused software not meeting the applicable requirements related to the aspects as specified in clauses 6.2.7.2 to 6.2.7.6.

EXPECTED OUTPUT: Software reuse file [DJF, SRF; SRR, PDR].

6.2.7.8

a. Reverse engineering techniques shall be applied to generate missing documentation and to reach the required verification and validation coverage.

b. For software products whose life cycle data from previous development are not available and reverse engineering techniques are not fully applicable, the following methods shall be applied:

1. generation of validation and verification documents based on the available user documentation (e.g. user manual) and execution of tests in order to achieve the required level of test coverage;

2. use of the product service history to provide evidence of the product's suitability for the current application, including information about:

(a) relevance of the product service history for the new operational environment;

(b) configuration management and change control of the software product;

(c) effectiveness of problem reporting;

(d) actual error rates and maintenance records;

(e) impact of modifications.

EXPECTED OUTPUT: Software reuse file [DJF, SRF; SRR, PDR].

6.2.7.9

a. The software reuse file shall be updated at project milestones to reflect the results of the identified corrective actions for reused software not meeting the project requirements.

EXPECTED OUTPUT: Software reuse file [DJF, SRF; CDR, QR, AR].

6.2.7.10

a. All the reused software shall be kept under configuration control.

6.2.7.11

a. The detailed configuration status of the reused software baseline shall be provided to the customer in the reuse file for acceptance.

EXPECTED OUTPUT: Software reuse file [DJF, SRF; SRR, PDR, CDR, QR, AR].

6.2.8 Automatic code generation

6.2.8.1

a. For the selection of tools for automatic code generation, the supplier shall evaluate the following aspects:

1. evolution of the tools in relation to the tools that use the generated code as an input;
2. customization of the tools to comply with project standards;
3. portability requirements for the generated code;
4. collection of the required design and code metrics;
5. verification of software components containing generated code;
6. configuration control of the tools including the parameters for customisation;
7. compliance with open standards.

NOTE Examples for item 1: compilers or code management systems.

6.2.8.2

a. The requirements on testing applicable to the automatically generated code shall ensure the achievement of the same objectives as those for manually generated code.

EXPECTED OUTPUT: *Validation and testing documentation [DJF, SValP; PDR], [DJF, SVS; CDR, QR, AR], [DJF, SUITP; PDR, CDR].*

6.2.8.3

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

6.2.8.4

a. Modelling standards for automatic code generation tools shall be defined and applied.

EXPECTED OUTPUT: *Modelling standards [PAF, -; SRR, PDR].*

6.2.8.5

a. Adherence to modelling standards shall be verified.

EXPECTED OUTPUT: *Software product assurance reports [PAF, -; -].*

6.2.8.6

a. Clause 6.3.4 shall apply to automatically generated code, unless the supplier demonstrates that the automatically generated code does not need to be manually modified.

6.2.8.7

a. The verification and validation documentation shall address separately the activities to be performed for manually and automatically generated code.

EXPECTED OUTPUT: *Validation and testing documentation [DJF, SValP; PDR], [DJF, SVS; CDR, QR, AR], [DJF, SUITP; PDR, CDR].*

6.3 Requirements applicable to individual software engineering processes or activities

6.3.1 Software related system requirements process

6.3.1.1

a. For the definition of the software related system requirements to be specified in the requirements baseline, ECSS-E-ST-40 clause 5.2 shall apply.

6.3.1.2

a. The requirements baseline shall be subject to documentation control and configuration management as part of the development documentation.

6.3.1.3

a. For the definition of the requirements baseline, all results from the safety and dependability analyses (including results from the HSIA ECSS-Q-ST-30 clause 6.4.2.3) shall be used.

6.3.2 Software requirements analysis

6.3.2.1

a. The requirements baseline shall be analyzed to fully and unambiguously define the software requirements in the technical specification.

6.3.2.2

a. The technical specification shall be subject to documentation control and configuration management as part of the development documentation.

6.3.2.3

a. For the definition of the technical specification, all results from the safety and dependability analyses (including results from the HSIA ECSS-Q-ST-30 clause 6.4.2.3) shall be used.

6.3.2.4

a. In addition to the functional requirements, the technical specification shall include all non-functional requirements necessary to satisfy the requirements baseline, including, as a minimum, the following:

1. performance,
2. safety,
3. reliability,
4. robustness,
5. quality,
6. maintainability,
7. configuration management,
8. security,
9. privacy,
10. metrication, and
11. verification and validation.

NOTE Performance requirements include requirements on numerical accuracy.

EXPECTED OUTPUT: *Software requirements specification [TS, SRS; PDR].*

6.3.2.5

a. Prior to the technical specification elaboration, customer and supplier shall agree on the following principles and rules as a minimum:

1. assignment of persons (on both sides) responsible for establishing the technical specification;
2. methods for agreeing on requirements and approving changes;
3. efforts to prevent misunderstandings such as definition of terms, explanations of background of requirements;
4. recording and reviewing discussion results on both sides.

6.3.3 Software architectural design and design of software items

6.3.3.1

a. The design definition file shall be subject to documentation control and configuration management.

6.3.3.2

a. Mandatory and advisory design standards shall be defined and applied.

EXPECTED OUTPUT: *Design standards [PAF, -; SRR, PDR].*

6.3.3.3

a. For software in which numerical accuracy is relevant to mission success specific rules on design and code shall be defined to ensure that the specified level of accuracy is obtained.

NOTE For example: for an attitude and orbit control subsystem, scientific data generation components.

EXPECTED OUTPUT: *Software product assurance plan [PAF, SPAP; PDR].*

6.3.3.4

a. Adherence to design standards shall be verified.

EXPECTED OUTPUT: *Software product assurance reports [PAF, -; -].*

6.3.3.5

a. The supplier shall define means, criteria and tools to ensure that the complexity and modularity of the design meet the quality requirements.

b. The design evaluation shall be performed in parallel with the design process, in order to provide feedback to the software design team.

EXPECTED OUTPUT: *Software product assurance plan [PAF, SPAP; PDR].*

6.3.3.6

a. Synthesis of the results obtained in the software complexity and modularity evaluation and corrective actions implemented shall be described in the software product assurance reports.

EXPECTED OUTPUT: *Software product assurance reports [PAF, -; -].*

6.3.3.7

a. The supplier shall review the design documentation to ensure that it contains the appropriate level of information for maintenance activities.

EXPECTED OUTPUT: *The following outputs are expected:*

a. Software product assurance plan [PAF, SPAP; PDR];

b. Software product assurance reports [PAF, -; -].

6.3.4 Coding

6.3.4.1

a. Coding standards (including consistent naming conventions and adequate commentary rules) shall be specified and observed.

EXPECTED OUTPUT: *Coding standards [PAF, -; PDR].*

6.3.4.2

a. The standards shall be consistent with the product quality requirements.

NOTE Coding standards depend on the software quality objectives (see clause 5.2.7).

EXPECTED OUTPUT: *Coding standards [PAF, -; PDR].*

6.3.4.3

a. The tools to be used in implementing and checking conformance with coding standards shall be identified in the product assurance plan before coding activities start.

EXPECTED OUTPUT: *Software product assurance plan [PAF, SPAP; PDR].*

6.3.4.4

a. Coding standards shall be reviewed with the customer to ensure that they reflect product quality requirements.

EXPECTED OUTPUT: *Coding standards and description of tools [PAF, -; PDR].*

6.3.4.5

a. Use of low-level programming languages shall be justified.

EXPECTED OUTPUT: *Software development plan [MGT, SDP; PDR].*

6.3.4.6

a. The supplier shall define measurements, criteria and tools to ensure that the software code meets the quality requirements.

EXPECTED OUTPUT: *Software product assurance plan [PAF, SPAP; PDR].*

b. The code evaluation shall be performed in parallel with the coding process, in order to provide feedback to the software programmers.

6.3.4.7

a. Synthesis of the code analysis results and corrective actions implemented shall be described in the software product assurance reports.

EXPECTED OUTPUT: *Software product assurance reports [PAF, -; -].*

6.3.4.8

a. The code shall be put under configuration control immediately after successful unit testing.

6.3.5 Testing and validation

6.3.5.1

a. Testing shall be performed in accordance with a strategy for each testing level (i.e. unit, integration, validation against the technical specification, validation against the requirements baseline, acceptance), which includes:

1. the types of tests to be performed;
2. the tests to be performed in accordance with the plans and procedures;
3. the means and organizations to perform assurance function for testing and validation.

NOTE For examples for item 1 are: functional, boundary, performance, and usability tests.

EXPECTED OUTPUT: *Software product assurance plan [PAF, SPAP; PDR, CDR].*

6.3.5.2

a. Based on the criticality of the software, test coverage goals for each testing level shall be agreed between the customer and the supplier and their achievement monitored by metrics:

1. for unit level testing;
2. for integration level testing;
3. for validation against the technical specification and validation against the requirements baseline.

EXPECTED OUTPUT: *Software product assurance plan [PAF, SPAP; PDR, CDR].*

6.3.5.3

a. The supplier shall ensure through internal review that the test procedures and data are adequate, feasible and traceable and that they satisfy the requirements.

EXPECTED OUTPUT: *Software product assurance reports [PAF, -; -].*

6.3.5.4

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

EXPECTED OUTPUT: *Test readiness review reports [DJF, -; TRR].*

6.3.5.5

a. Test coverage shall be checked with respect to the stated goals.

EXPECTED OUTPUT: *Software product assurance reports [PAF, -; -].*

b. Feedback from the results of test coverage evaluation shall be continuously provided to the software developers.

6.3.5.6

a. The supplier shall ensure that nonconformances and software problem reports detected during testing are properly documented and reported to those concerned.

EXPECTED OUTPUT: *Nonconformance reports and software problem reports [DJF, -; CDR, QR, AR, ORR].*

6.3.5.7

a. The test coverage of configurable code shall be checked to ensure that the stated requirements are met in each tested configuration.

EXPECTED OUTPUT: *Statement of compliance with test plans and procedures [PAF, -; CDR, QR, AR, ORR].*

6.3.5.8

a. The completion of actions related to software problem reports generated during testing and validation shall be verified and recorded.

EXPECTED OUTPUT: *Software problem reports [DJF, -; SRR, PDR, CDR, QR, AR, ORR].*

6.3.5.9

a. Provisions shall be made to allow witnessing of tests by the customer.

6.3.5.10

a. Provisions shall be made to allow witnessing of tests by supplier personnel independent of the development.

NOTE For example: specialist software product assurance personnel.

6.3.5.11

a. The supplier shall ensure that:

1. tests are conducted in accordance with approved test procedures and data,
2. the configuration under test is correct,
3. the tests are properly documented, and
4. the test reports are up to date and valid.

EXPECTED OUTPUT: *Statement of compliance with test plans and procedures [PAF, -; CDR, QR, AR, ORR].*

6.3.5.12

a. The supplier shall ensure that tests are repeatable by verifying the storage and recording of tested software, support software, test environment, supporting documents and problems found.

EXPECTED OUTPUT: *Software product assurance reports [PAF, -; -].*

6.3.5.13

a. The supplier shall confirm in writing that the tests are successfully completed.

EXPECTED OUTPUT: *Testing and validation reports [DJF, -; CDR, QR, AR, ORR].*

6.3.5.14

a. Review boards looking to engineering and product assurance aspects shall be convened after the completion of test phases, as defined in the software development plan.

6.3.5.15

a. Areas affected by any modification shall be identified and re-tested (regression testing).

6.3.5.16

a. In case of re-testing, all test related documentation (test procedures, data and reports) shall be updated accordingly.

EXPECTED OUTPUT: *Updated test documentation [DJF, -; CDR, QR, AR, ORR].*

6.3.5.17

a. The need for regression testing and additional verification of the software shall be analysed after any change of the platform hardware.

EXPECTED OUTPUT: *Updated test documentation [DJF, -; CDR, QR, AR, ORR].*

6.3.5.18

a. The need for regression testing and additional verification of the software shall be analysed after a change or update of any tool used to generate it.

NOTE For example: source code or object code.

EXPECTED OUTPUT: *Updated test documentation [DJF, -; CDR, QR, AR, ORR].*

6.3.5.19

a. Validation shall be carried out by staff who have not taken part in the design or coding of the software being validated.

NOTE This can be achieved at the level of the whole software product, or on a component by component basis.

6.3.5.20

a. Validation of the flight software against the requirement baseline on the flight equipment model shall be performed on a software version without any patch.

6.3.5.21

a. The supplier shall review the test documentation to ensure that it is up to date and organized to facilitate its reuse for maintenance.

6.3.5.22

a. Tests shall be organized as activities in their own right in terms of planning, resources and team composition.

EXPECTED OUTPUT: *Test and validation documentation [DJF, SValP; PDR], [DJF, SUITP; PDR, CDR].*

6.3.5.23

a. The necessary resources for testing shall be identified early in the life cycle, taking into account the operating and maintenance requirements.

EXPECTED OUTPUT: *Test and validation documentation [DJF, SValP; PDR], [DJF, SUITP; PDR, CDR].*

6.3.5.24

a. Test tool development or acquisition (hardware and software) shall be planned for in the overall project plan.

EXPECTED OUTPUT: *Test and validation documentation [DJF, SValP; PDR], [DJF, SUITP; PDR, CDR].*

6.3.5.25

a. The supplier shall establish and review the test procedures and data before starting testing activities and also document the constraints of the tests concerning physical, performance, functional, controllability and observability limitations.

EXPECTED OUTPUT: *Test and validation documentation [DJF, SValP; PDR], [DJF, SVS; CDR, QR, AR], [DJF, SUITP; PDR, CDR].*

6.3.5.26

a. Before offering the product for delivery and customer acceptance, the supplier shall validate its operation as a complete product, under conditions similar to the application environment as specified in the requirements baseline.

6.3.5.27

a. When testing under the operational environment is performed, the following concerns shall be addressed:

1. the features to be tested in the operational environment;
2. the specific responsibilities of the supplier and customer for carrying out and evaluating the test;
3. restoration of the previous operational environment (after test).

EXPECTED OUTPUT: *Test and validation documentation [DJF, -; AR].*

6.3.5.28

a. Independent software validation shall be performed by a third party.

NOTE This requirement is applicable where the risks associated with the project justify the costs involved. The customer can consider a less rigorous level of independence, e.g. an independent team in the same organization.

EXPECTED OUTPUT: *The following outputs are expected:*

a. ISVV plan [DJF, -; SRR, PDR];

b. ISVV report [DJF, -; PDR, CDR, QR, AR].

6.3.5.29

a. The validation shall include testing in the different configurations possible or in a representative set of them when it is evident that the number of possible configurations is too high to allow validation in all of them.

EXPECTED OUTPUT: *Test and validation documentation [DJF, SValP; PDR], [DJF, SVS; CDR, QR, AR].*

6.3.5.30

a. Software containing deactivated code shall be validated specifically to ensure that the deactivated code cannot be activated or that its accidental activation cannot harm the operation of the system.

EXPECTED OUTPUT: *Testing and validation reports [DJF, -; CDR, QR, AR].*

6.3.5.31

a. Software containing configurable code shall be validated specifically to ensure that unintended configuration cannot be activated at run time or included during code generation.

EXPECTED OUTPUT: *Testing and validation reports [DJF, -; CDR, QR, AR].*

6.3.5.32

a. Activities for the validation of the quality requirements shall be specified in the definition of the validation specification.

EXPECTED OUTPUT: *Software validation specification [DJF, SVS; CDR, QR, AR]*.

6.3.6 Software delivery and acceptance

6.3.6.1

a. The roles, responsibilities and obligations of the supplier and customer during installation shall be established.

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

6.3.6.2

a. The installation shall be performed in accordance with the installation procedure.

6.3.6.3

a. The customer shall establish an acceptance test plan specifying the intended acceptance tests including specific tests suited to the target environment (see ECSS-E-ST-40 clause 5.7.3.1).

NOTE 1 The acceptance tests can be partly made up of tests used during previous test activities.

NOTE 2 The acceptance test plan takes into account the requirement for operational demonstration, either as part of acceptance or after acceptance.

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

6.3.6.4

a. The customer shall ensure that the acceptance tests are performed in accordance with the approved acceptance test plan (see ECSS-E-ST-40 clause 5.7.3.2).

6.3.6.5

a. Before the software is presented for customer acceptance, the supplier shall ensure that:

1. the delivered software complies with the contractual requirements (including any specified content of the software acceptance data package);
2. the source and object code supplied correspond to each other;
3. all agreed changes are implemented;
4. all nonconformances are either resolved or declared.

6.3.6.6

a. The customer shall verify that the executable code was regenerated from configuration managed source code components and installed in accordance with predefined procedures on the target environment.

6.3.6.7

a. Any discovered problems shall be documented in nonconformance reports.

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

6.3.6.8

a. On completion of the acceptance tests, a report shall be drawn up and be signed by the supplier's representatives, the customer's representatives, the software quality engineers of both parties and the representative of the organization charged with the maintenance of the software product.

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

6.3.6.9

a. The customer shall certify conformance to the procedures and state the conclusion concerning the test result for the software product under test (accepted, conditionally accepted, rejected).

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

6.3.7 Operations

6.3.7.1

a. During operations, the quality of the mission products related to software shall be agreed with the customer and users.

NOTE Quality of mission products can include parameters such as: error-free data, availability of data and permissible outages; permissible information degradation.

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

6.3.7.2

a. During the demonstration that the software conforms to the operational requirements, the following shall be covered as a minimum:

1. availability and maintainability of the host system (including reboot after maintenance interventions);
2. safety features;
3. human-computer interface;
4. operating procedures;
5. ability to meet the mission product quality requirements.

EXPECTED OUTPUT: *Validation of the operational requirements [PAF, -; ORR].*

6.3.7.3

a. The product assurance plan for system operations shall include consideration of software.

EXPECTED OUTPUT: *Input to product assurance plan for systems operation [PAF, -; ORR]*

6.3.8 Maintenance

6.3.8.1

a. The organization responsible for maintenance shall be identified to allow a smooth transition into the operations and maintenance.

NOTE An organization, with representatives from both supplier and customer, can be set up to support the maintenance activities. Attention is drawn to the importance of the flexibility of this organization to cope with the unexpected occurrence of problems and the identification of facilities and resources to be used for the maintenance activities.

EXPECTED OUTPUT: *Maintenance plan [MF, -; QR, AR, ORR].*

6.3.8.2

a. The maintenance organization shall specify the assurance, verification and validation activities applicable to maintenance interventions.

EXPECTED OUTPUT: *Maintenance plan [MF, -; QR, AR, ORR].*

6.3.8.3

a. The maintenance plans shall be verified against specified requirements for maintenance of the software product.

NOTE The maintenance plans and procedures can address corrective, improving, adaptive and preventive maintenance, differentiating between "routine" and "emergency" maintenance activities.

6.3.8.4

a. The maintenance plans and procedures shall include the following as a minimum:

1. scope of maintenance;
2. identification of the first version of the software product for which maintenance is to be done;
3. support organization;
4. maintenance life cycle;
5. maintenance activities;

6. quality measures to be applied during the maintenance;

7. maintenance records and reports.

EXPECTED OUTPUT: Maintenance plan [MF, -; QR, AR, ORR].

6.3.8.5

a. Rules for the submission of maintenance reports shall be established and agreed as part of the maintenance plan.

EXPECTED OUTPUT: Maintenance plan [MF, -; QR, AR, ORR].

6.3.8.6

a. All maintenance activities shall be logged in predefined formats and retained.

EXPECTED OUTPUT: Maintenance records [MF, -; -].

6.3.8.7

a. Maintenance records shall be established for each software product, including, as a minimum, the following information,:

1. list of requests for assistance or problem reports that have been received and the current status of each;

2. organization responsible for responding to requests for assistance or implementing the appropriate corrective actions;

3. priorities assigned to the corrective actions;

4. results of the corrective actions;

5. statistical data on failure occurrences and maintenance activities.

NOTE The record of the maintenance activities can be utilized for evaluation and enhancement of the software product and for improvement of the quality system itself.

EXPECTED OUTPUT: Maintenance records [MF, -; -].

7
Software product quality assurance

7.1 Product quality objectives and metrication

7.1.1 Deriving of requirements

a. The software quality requirements (including safety and dependability requirements) shall be derived from the requirements defined at system level.

EXPECTED OUTPUT: *The following outputs are expected:*

a. Requirement baseline [RB, SSS; SRR];

b. Technical specification [TS, SRS; PDR].

7.1.2 Quantitative definition of quality requirements

a. Quality requirements shall be expressed in quantitative terms or constraints.

EXPECTED OUTPUT: *The following outputs are expected:*

a. Requirement baseline [RB, SSS; SRR];

b. Technical specification [TS, SRS; PDR].

7.1.3 Assurance activities for product quality requirements

a. The supplier shall define assurance activities to ensure that the product meets the quality requirements as specified in the technical specification.

EXPECTED OUTPUT: *Software product assurance plan [PAF, SPAP; SRR, PDR].*

7.1.4 Product metrics

a. In order to verify the implementation of the product quality requirements, the supplier shall define a metrication programme based on the identified quality model (see clause 5.2.7), specifying:

1. the metrics to be collected and stored;
2. the means to collect metrics (measurements);

3. the target values, with reference to the product quality requirements;

4. the analyses to be performed on the collected metrics, including the ones to derive:

   (a) descriptive statistics;

   (b) trend analysis (such as trends in software problems).

5. how the results of the analyses performed on the collected metrics are fed back to the development team and used to identify corrective actions;

6. the schedule of metrics collection, storing, analysis and reporting, with reference to the whole software life cycle.

NOTE 1 Guidance for software metrication programme implementation can be found in ECSS-Q-HB-80-04.

NOTE 2 Example to item 4(a): the number of units at each level of complexity.

EXPECTED OUTPUT: *Software product assurance plan [PAF, SPAP; SRR, PDR].*

7.1.5 Basic metrics

a. The following basic products metrics shall be used:

1. size (code);

2. complexity (design, code);

3. fault density and failure intensity;

4. test coverage;

5. number of failures.

EXPECTED OUTPUT: *Software product assurance plan [PAF, SPAP; SRR, PDR].*

7.1.6 Reporting of metrics

a. The results of metrics collection and analysis shall be included in the software product assurance reports, in order to provide the customer with an insight into the level of quality obtained.

EXPECTED OUTPUT: *Software product assurance reports [PAF, -; -].*

7.1.7 Numerical accuracy

a. Numerical accuracy shall be estimated and verified.

EXPECTED OUTPUT: *Numerical accuracy analysis [DJF, SVR; PDR, CDR, QR].*

7.1.8 Analysis of software maturity

a. The supplier shall define the organization and means implemented to collect and analyse data required for the study of software maturity.

NOTE For example: failures, corrections, duration of runs.

EXPECTED OUTPUT: *Software product assurance reports [PAF, -; -].*

7.2 Product quality requirements

7.2.1 Requirements baseline and technical specification

7.2.1.1

a. The software quality requirements shall be documented in the requirements baseline and technical specification.

EXPECTED OUTPUT: *The following outputs are expected:*

a. Requirement baseline [RB, SSS; SRR];

b. Technical specification [TS, SRS; PDR].

7.2.1.2

a. The software requirements shall be:

1. correct;
2. unambiguous;
3. complete;
4. consistent;
5. verifiable;
6. traceable.

7.2.1.3

a. For each requirement the method for verification and validation shall be specified.

EXPECTED OUTPUT: *The following outputs are expected:*

a. Requirement baseline [RB, SSS; SRR];

b. Technical specification [TS, SRS; PDR].

7.2.2 Design and related documentation

7.2.2.1

a. The software design shall meet the non-functional requirements as documented in the technical specification.

7.2.2.2

a. The software shall be designed to facilitate testing.

7.2.2.3

a. Software with a long planned lifetime shall be designed with minimum dependency on the operating system and the hardware, in order to aid portability.

NOTE This requirement is applicable to situations where the software lifetime can lead to the obsolescence and non-availability of the original operating system and/or hardware, thereby jeopardizing the maintainability the software.

EXPECTED OUTPUT: *The following outputs are expected:*

a. Software product assurance plan [PAF, SPAP; SRR, PDR];

b. Justification of design choices [DDF, SDD; PDR, CDR].

7.2.3 Test and validation documentation

7.2.3.1

a. Detailed test and validation documentation (data, procedures and expected results) defined in the ECSS-E-ST-40 DJF shall be consistent with the defined test and validation strategy (see clause 6.3.5 and ECSS-E-ST-40 clauses 5.5.3, 5.5.4, 5.6 and 5.8).

7.2.3.2

a. The test documentation shall cover the test environment, tools and test software, personnel required and associated training requirements.

7.2.3.3

a. The criteria for completion of each test and any contingency steps shall be specified.

7.2.3.4

a. Test procedures, data and expected results shall be specified.

7.2.3.5

a. The hardware and software configuration shall be identified and documented as part of the test documentation.

7.2.3.6

a. For any requirements not covered by testing a verification report shall be drawn up documenting or referring to the verification activities performed.

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

7.3 Software intended for reuse

7.3.1 Customer requirements

a. For the development of software intended for reuse, ECSS-E-ST-40 clauses 5.2.4.7 and 5.4.3.6 shall apply.

7.3.2 Separate documentation

a. The information related to the components developed for reuse shall be separated from the others in the technical specification, design justification file, design definition file and product assurance file.

7.3.3 Self-contained information

a. The information related to components developed for reuse in the technical specification, the design justification file, the design definition file and the product assurance file shall be self-contained.

7.3.4 Requirements for intended reuse

a. The technical specification of components developed for reuse shall include requirements for maintainability, portability and verification of those components.

EXPECTED OUTPUT: *Technical specification for reusable components [TS, -; PDR].*

7.3.5 Configuration management for intended reuse

a. The configuration management system shall include provisions for handling specific aspects of software developed for reuse, such as:

1. longer lifetime of the components developed for reuse compared to the other components of the project;

2. evolution or change of the development environment for the next project that intends to use the components;

3. transfer of the configuration and documentation management information to the next project reusing the software.

EXPECTED OUTPUT: *Software configuration management plan [MGT, SCMP; SRR, PDR].*

7.3.6 Testing on different platforms

a. Where the components developed for reuse are developed to be reusable on different platforms, the testing of the software shall be performed on all those platforms.

EXPECTED OUTPUT: *Verification and validation documentation for reusable components [DJF, -; CDR].*

7.3.7 Certificate of conformance

a. The supplier shall provide a certificate of conformance that the tests have been successfully completed on all the relevant platforms.

NOTE In case not all platforms are available, the certificate of conformance states the limitations of the validation performed.

EXPECTED OUTPUT: *Verification and validation documentation for reusable components [DJF, -; CDR].*

7.4 Standard ground hardware and services for operational system

7.4.1 Hardware procurement

a. The subcontracting and procurement of hardware shall be carried out according to the requirements of ECSS-Q-ST-20 clause 5.4.

EXPECTED OUTPUT: *The following outputs are expected:*

a. Justification of selection of operational ground equipment [DJF, -; SRR, PDR];

b. Receiving inspection reports [PAF, -; SRR, PDR].

7.4.2 Service procurement

a. The procurement of support services to be used in operational phases shall be justified as covering service level agreements, quality of services and escalation procedures, as needed for system exploitation and maintenance.

EXPECTED OUTPUT: *Justification of selection of operational support services [DJF, -; SRR, PDR].*

7.4.3 Constraints

a. The choice of procured hardware and services shall address the constraints associated with both the development and the actual use of the software.

EXPECTED OUTPUT: *Justification of selection of operational ground equipment [DJF, -; SRR, PDR].*

7.4.4 Selection

a. The ground computer equipment and supporting services for implementing the final system shall be selected according to the project requirements regarding:

1. performance;
2. maintenance;
3. durability and technical consistency with the operational equipment;
4. the assessment of the product with respect to requirements, including the criticality category;
5. the available support documentation;
6. the acceptance and warranty conditions;
7. the conditions of installation, preparation, training and use;
8. the maintenance conditions, including the possibilities of evolutions;
9. copyright constraints;
10. availability;
11. compatibility;
12. site operational constraints.

EXPECTED OUTPUT: *Justification of selection of operational ground equipment [DJF, -; SRR, PDR].*

7.4.5 Maintenance

a. Taking account of the provider's maintenance and product policy, it shall be ensured that the hardware and support services can be maintained throughout the specified life of the software product within the operational constraints.

7.5 Firmware

7.5.1 Device programming

a. The supplier shall establish procedures for firmware device programming and duplication of firmware devices.

EXPECTED OUTPUT: *Software product assurance plan [PAF, SPAP; PDR].*

7.5.2 Marking

a. The firmware device shall be indelibly marked to allow the identification (by reference) of the hardware component and of the software component.

EXPECTED OUTPUT: *Software product assurance plan [PAF, SPAP; PDR].*

7.5.3 Calibration

a. The supplier shall ensure that the firmware programming equipment is calibrated.

Annex A (informative) Software documentation

This annex defines the structure of the software documents to be produced, as depicted in Figure A-1.

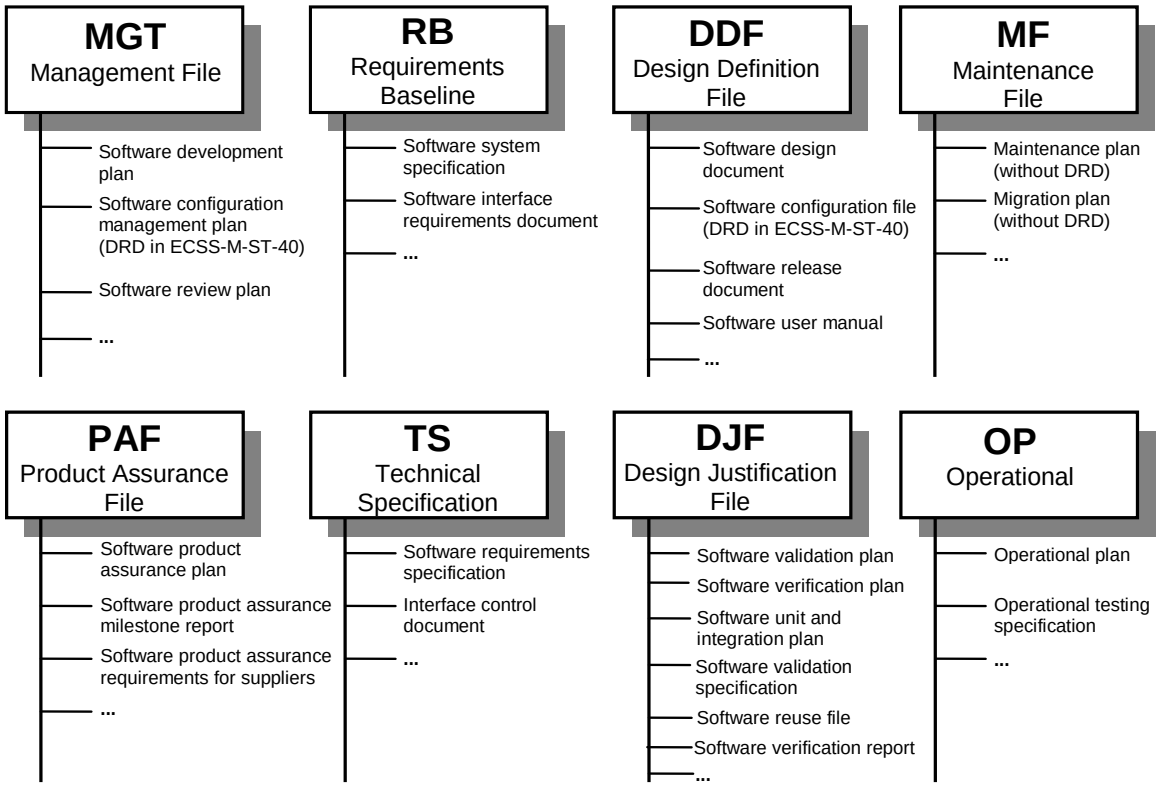

Figure A-1: Overview of software documents

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-E-ST-40.

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	-	✓	✓				
	Software integration test plan (SUITP)	ECSS-E-ST-40C Annex K		✓	✓			

Related file	DRL item (e.g. Plan, document, file, report, form, matrix)	DRL item having a DRD	SRR	PDR	CDR	QR	AR	ORR
	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 problem reports and nonconformance reports	-	✓	✓	✓	✓	✓	✓
	Joint review report	-	✓	✓	✓	✓	✓	
	Justification of selection of operational ground equipment and services	-	✓	✓				

Related file	DRL item (e.g. Plan, document, file, report, form, matrix)	DRL item having a DRD	SRR	PDR	CDR	QR	AR	ORR
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 members	-	✓					
	Procurement data	-	✓	✓				
MF	Maintenance plan	-				✓	✓	✓
	Maintenance records	-				✓	✓	✓
	SPR and NCR - Modification analysis report - Problem analysis report - Modification identification	-						
	Migration plan and notification	-						
	Retirement plan and notification	-						
OP	Software operation support plan	-						✓
	Operational testing results	-						✓
	SPR and NCR - User's request record software product - 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
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 procedures	-		✓				
	Software dependability and safety analysis report - Criticality classification of software components	-		✓	✓	✓	✓	
	Software product assurance reports	-						
	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	-						
	(Preliminary) alert information	-						
	Result of pre-award audits and assessments, and of procurement sources	-						
	Software process assessment plan	-						
	Software process assessment records	-						
	Review and inspection reports	-						

Related file	DRL item (e.g. Plan, document, file, report, form, matrix)	DRL item having a DRD	SRR	PDR	CDR	QR	AR	ORR
	Receiving inspection reports	-	✓	✓	✓	✓		
	Input to product assurance plan for systems operation	-						✓

Annex B (normative) Software product assurance plan (SPAP) - DRD

B.1 DRD identification

B.1.1 Requirement identification and source document

The software product assurance plan (SPAP) is called from the normative provisions summarized in Table B-1.

Table B-1: SPAP traceability to ECSS-E-ST-40 and ECSS-Q-ST-80 clauses

ECSS Standard	Clause	DRD section
ECSS-Q-ST-80	5.1.2.1	<5.1>.a, <5.1>.b
	5.1.2.2	<5.1>.a, <5.1>.b
	5.1.2.3	<5.1>.b
	5.1.3.1	<5.3>
	5.1.4.1	<5.1>.b
	5.2.1.1	All
	5.2.1.3	All
	5.2.1.4	<5.10>
	5.2.1.5	<8>
	5.2.6.1c	<6.4>
	5.2.7.2	<5.5>
	5.4.3.3	All
	5.4.3.4	All
	5.6.1.1	<5.8>
	6.1.1	<6.1>
	6.1.5	<6.1>.c
	6.2.1.4	<6.2>
	6.2.3.2	<6.3>.c
	6.2.3.4	<6.3>.c
	6.2.3.5	<6.3>.c

ECSS Standard	Clause	DRD section
	6.2.4.8	<6.4>.d
	6.2.4.9	<6.4>.d
	6.2.4.11	<6.4>.d
	6.2.5.1	<6.5>.e
	6.2.5.2	<6.5>.e
	6.2.7.2	<6.6>.f
	6.2.7.3	<6.6>.f
	6.2.7.4	<6.6>.f
	6.2.7.5	<6.6>.f
	6.3.3.3	<6.7>.a.2.h.3
	6.3.3.5	<6.7>.a.2.g.2
	6.3.3.7	<6.7>.a.2.g.2
	6.3.4.3	<6.7>.a.3.h.2
	6.3.4.6	<6.7>.a.3.g.3
	6.3.5.1	<6.7>.a.4.g.4
	6.3.5.2	<6.7>.a.4.g.4
	7.1.3	<7.b.6>.b.4
	7.1.5	<7>.b.1
	7.1.6	<7>.b.1
	7.2.2.3	<7>.a
	7.5.1	<6.8>.c.19.h.3
	7.5.2	<6.8>.c.19.h.3

B.1.2 Purpose and objective

The software product assurance plan is a constituent of the product assurance file (PAF).

The purpose of the software product assurance plan is to provide information on the organizational aspects and the technical approach to the execution of the software product assurance programme

B.2 Expected response

B.2.1 Scope and content

<1> Introduction

a. The SPAP shall contain a description of the purpose, objective, content and the reason prompting its preparation.

<2> Applicable and reference documents

a. The SPAP shall list the applicable and reference documents to support the generation of the document.

<3> Terms, definitions and abbreviated terms

a. The SPAP shall include any additional terms, definition or abbreviated terms used.

<4> System Overview

a. The SPAP shall include or refer to a description of the system and software products being developed.

<5> Software product assurance programme implementation

<5.1> Organization

a. The SPAP shall describe the organization of software product assurance activities, including responsibility, authority and the interrelation of personnel who manage, perform and verify work affecting software quality.

b. The following topics shall be included:

1. organizational structure;
2. interfaces of each organisation, either external or internal, involved in the project;
3. relationship to the system level product assurance and safety;
4. independence of the software product assurance function;
5. delegation of software product assurance tasks to a lower level supplier, if any.

<5.2> Responsibilities

a. The SPAP shall describe the responsibilities of the software product assurance function.

<5.3> Resources

a. The SPAP shall describe the resources to be used to perform the software product assurance function.

b. The description in B.2.1<5.3>a. shall include human resources and skills, hardware and software tools.

<5.4> Reporting

a. The SPAP shall describe the reporting to be performed by software product assurance.

<5.5> Quality models

a. The SPAP shall describe the quality models applicable to the project and how they are used to specify the quality requirements.

<5.6> Risk management

a. The SPAP shall describe the contribution of the software product assurance function to the project risk management.

<5.7> Supplier selection and control

a. The SPAP shall describe the contribution of the software product assurance function to the next level suppliers selection and control.

<5.8> Methods and tools

a. The SPAP shall describe the methods and tools used for all the activities of the development cycle, and their level of maturity.

<5.9> Process assessment and improvement

a. The SPAP shall state the scope and objectives of process assessment.

b. The SPAP shall describe the methods and tools to be used for process assessment and improvement.

<5.10> Operations and maintenance (optional)

a. The SPAP shall specify the quality measures related to the operations and maintenance processes (alternatively, a separate SPAP is produced).

<6> Software process assurance

<6.1> Software development cycle

a. The SPAP shall refer to the software development cycle description in the software development plan.

b. If not covered in the software development plan, the life cycle shall be described.

c. The life cycle shall include a milestone immediately before the starting of the software validation.

<6.2> Projects plans

a. The SPAP shall describe all plans to be produced and used in the project.

b. The relationship between the project plans and a timely planning for their preparation and update shall be described.

<6.3> Software dependability and safety

a. The SPAP shall contain a description and justification of the measures to be applied for the handling of critical software, including the analyses to be performed and the standards applicable for critical software.

<6.4> Software documentation and configuration management

a. The SPAP shall describe the contribution of the software product assurance function to the proper implementation of documentation and configuration management.

b. The nonconformance control system shall be described or referenced. The point in the software life cycle from which the nonconformance procedures apply shall be specified.

c. The SPAP shall identify method and tool to protect the supplied software, a checksum-type key calculation for the delivered operational software, and a labelling method for the delivered media.

<6.5> Process metrics

a. The SPAP shall describe the process metrics derived from the defined quality models, the means to collect, store and analyze them, and the way they are used to manage the development processes.

<6.6> Reuse of software

a. The SPAP shall describe the approach for the reuse of existing software, including delta qualification.

<6.7> Product assurance planning for individual processes and activities

a. The following processes and activities shall be covered, taking into account the project scope and life cycle:
   1. software requirements analysis;
   2. software architectural design and design of software items;
   3. coding;
   4. testing and validation (including regression testing);
   5. verification;
   6. software delivery and acceptance;
   7. operations and maintenance.

<6.8> Procedures and standards

a. The SPAP shall describe or list by reference all procedures and standards applicable to the development of the software in the project.

b. The software product assurance measures to ensure adherence to the project procedures and standards shall be described.

c. The standards and procedures to be described or listed in accordance with B.2.1<6.8>a shall be as a minimum those covering the following aspects:

1. project management;
2. risk management;
3. configuration and documentation management;
4. verification and validation;
5. requirements engineering;
6. design;
7. coding;
8. metrication;
9. nonconformance control;
10. audits;
11. alerts;
12. procurement;
13. reuse of existing software;
14. use of methods and tools;
15. numerical accuracy;
16. delivery, installation and acceptance;
17. operations;
18. maintenance;
19. device programming and marking.

<7> Software product quality assurance

a. The SPAP shall describe the approach taken to ensure the quality of the software product.

b. The description of the approach specified in B.2.1<7>a shall include the:

1. specification of the product metrics, their target values and the means to collect them;
2. definition of a timely metrication programme;
3. analyses to be performed on the collected metrics;
4. way the results are fed back to the development team;
5. documentation quality requirements;
6. assurance activities meant to ensure that the product meets the quality requirements.

<8> Compliance matrix to software product assurance requirements

a. The SPAP shall include the compliance matrix to the applicable software product assurance requirements (e.g. ECSS-Q-ST-80 clauses, as tailored by a product assurance requirements document), or provide a reference to it.

b. For each software product assurance requirement, the following information shall be provided:

1. requirement identifier;
2. compliance
   (C = compliant, NC = non–compliant, NA = not applicable);
3. reference to the project documentation covering the requirement (e.g. section of the software product assurance plan);
4. remarks.

B.2.2 Special remarks

The response to this DRD may be combined with the response to the project product assurance plan, as defined in ECSS-Q-ST-10.

Annex C (normative) Software product assurance milestone report (SPAMR) - DRD

C.1 DRD identification

C.1.1 Requirement identification and source document

The software product assurance milestone report (SPAMR) is called from the normative provisions summarized in Table C-1.

Table C-1: SPAMR traceability to ECSS-Q-ST-80 clauses

ECSS Standard	Clause	DRD section
ECSS-Q-ST-80	5.2.2.3	All
	5.6.1.2	<5>.a
	5.6.1.3	<5>.b
	6.2.5.4	<7>
	6.2.5.5	<7>
	6.2.6.3	<4>
	6.2.6.7	<4>
	6.2.8.5	<6>
	6.3.3.4	<6>
	6.3.3.6	<6>.a.1
	6.3.3.7	<6>.a.2
	6.3.4.7	<6>
	6.3.5.3	<8>
	6.3.5.5	<8>
	6.3.5.12	<8>
	7.1.6	<7>
	7.1.8	<7>

C.1.2 Purpose and objective

The software product assurance milestone report is a constituent of the product assurance file (PAF).

The main purpose of the software product assurance milestone report is to collect and present at project milestones the reporting on the software product assurance activities performed during the past project phases.

C.2 Expected response

C.2.1 Scope and content

<1> Introduction

a. The SPAMR shall contain a description of the purpose, objective, content and the reason prompting its preparation.

<2> Applicable and reference documents

a. The SPAMR shall list the applicable and reference documents to support the generation of the document.

<3> Terms, definitions and abbreviated terms

a. The SPAMR shall include any additional terms, definition or abbreviated terms used.

<4> Verification activities performed

a. The SPAMR shall contain reporting on verification activities performed by the product assurance function, including:
   1. reviews;
   2. inspections;
   3. walk-throughs;
   4. review of traceability matrices;
   5. documents reviewed.

b. The SPAMR shall contain reporting on the verification of the measures applied for the handling of critical software.

<5> Methods and tools

a. The SPAMR shall include or reference a justification of the suitability of the methods and tools applied in all the activities of the development cycle, including requirements analysis, software specification, design, coding, validation, testing, configuration management, verification and product assurance.

b. The SPAMR shall include reporting on the correct use of methods and tools.

<6> Adherence to design and coding standards

a. The SPAMR shall include reporting on the adherence of software products to the applicable modelling, design and coding standards, including:

   1. reporting on the application of measures meant to ensure that the design complexity and modularity meet the quality requirements;

   2. reporting on design documentation w.r.t. suitability for maintenance.

<7> Product and process metrics

a. The SPAMR shall include reporting on the collected product and process metrics, the relevant analyses performed, the corrective actions undertaken and the status of these actions.

b. The results of the software maturity analysis shall also be reported.

<8> Testing and validation

a. The SPAMR shall include reporting on adequacy of the testing and validation documentation (including feasibility, traceability repeatability), and on the achieved test coverage w.r.t. stated goals.

<9> SPRs and SW NCRs

a. The SPAMR shall include reporting on the status of software problem reports and nonconformances relevant to software.

<10> References to progress reports

a. Whenever relevant and up-to-date information has been already delivered as part of the regular PA progress reporting, a representative summary shall be provided, together with a detailed reference to the progress report(s) containing that information.

C.2.2 Special remarks

The response to this DRD may be combined with the response to the project product assurance report, as defined in ECSS-Q-ST-10.

Annex D (normative) Tailoring of this Standard based on software criticality

D.1 Software criticality categories

Criticality categories are assigned to software products as specified in ECSS-Q-ST-30 clause 5.4, and ECSS-Q-ST-40 clause 6.5.6.3.

Table D-1 describes the relationship between the criticality category of the software products, the highest criticality of the functions implemented by the software and the existing system compensating provisions, as described in ECSS-Q-ST-30, clause 5.4, and ECSS-Q-ST-40, clause 6.5.6.3.

To any software product type described in the right column, the corresponding criticality category in the left column is assigned. E.g. both "Software involved in category I functions AND: no compensating provisions exist" and "Software included in compensating provisions for category I functions" are category A software.

For criticality classification of software components, clause 6.2.2 of this Standard applies.

Table D-1: Software criticality categories

Software criticality category	Definition
A	Software involved in category I functions AND: no compensating provisions exist
	Software included in compensating provisions for category I functions
B	Software involved in category I functions AND: at least one of the following compensating provisions is available, meeting the requirements defined in ECSS-Q-ST-30 clause 5.4 and ECSS-Q-ST-40 clause 6.5.6.3: - A hardware implementation - A software implementation; this software implementation shall be classified as criticality A - An operational procedure
	Software involved in category II functions AND: no compensating provisions exist
	Software included in compensating provisions for category II functions
C	Software involved in category II functions AND: at least one of the following compensating provisions is available, meeting the requirements defined in ECSS-Q-ST-30 clause 5.4 and ECSS-Q-ST-40 clause 6.5.6.3: - A hardware implementation - A software implementation; this software implementation shall be classified as criticality B - An operational procedure
	Software involved in category III functions AND: no compensating provisions exist
	Software included in compensating provisions for category III functions
D	Software involved in category III functions AND: at least one of the following compensating provisions is available, meeting the requirements defined in ECSS-Q-ST-30 clause 5.4 and ECSS-Q-ST-40 clause 6.5.6.3: - A hardware implementation - A software implementation; this software implementation shall be classified as criticality C - An operational procedure
	Software involved in category IV functions AND: no compensating provisions exist

D.2 Applicability matrix

The following applicability matrix represents a tailoring of the requirements of this Standard based on the software criticality categories defined as per D.1.

For each clause of this Standard and for each software criticality category, an indication is given whether that clause is applicable (Y), not applicable (N), or applicable under the conditions thereby specified to that software criticality category.

Table D-2: Applicability matrix based on software criticality

Clause	Description	A	B	C	D
5	**Software product assurance programme implementation**	-	-	-	-
5.1	Organization and responsibility	-	-	-	-
5.1.1	Organization	Y	Y	Y	Y
5.1.2	Responsibility and authority	-	-	-	-
5.1.2.1		Y	Y	Y	Y
5.1.2.2		Y	Y	Y	Y
5.1.2.3		Y	Y	Y	Y
5.1.3	Resources	-	-	-	-
5.1.3.1		Y	Y	Y	Y
5.1.3.2		Y	Y	Y	N
5.1.4	Software product assurance manager/engineer	-	-	-	-
5.1.4.1		Y	Y	Y	Y
5.1.4.2		Y	Y	Y	Y
5.1.5	Training	-	-	-	-
5.1.5.1		Y	Y	Y	Expected output not required
5.1.5.2		Y	Y	Y	N
5.1.5.3		Y	Y	Y	Y
5.1.5.4		Y	Y	Y	Y
5.2	Software product assurance programme management	-	-	-	-
5.2.1	Software product assurance planning and control	-	-	-	-
5.2.1.1		Y	Y	Y	Y
5.2.1.2		Y	Y	Y	Y
5.2.1.3		Y	Y	Y	Y
5.2.1.4		Y	Y	Y	Y
5.2.1.5		Y	Y	Y	Y
5.2.2	Software product assurance reporting	-	-	-	-

Clause	Description	A	B	C	D
5.2.2.1		Y	Y	Y	Y
5.2.2.2		Y	Y	Y	Y
5.2.2.3		Y	Y	Y	Y
5.2.3	Audits	Y	Y	Y	Audits planned and performed only when necessary
5.2.4	Alerts	Y	Y	Y	Y
5.2.5	Software problems	-	-	-	-
5.2.5.1		Y	Y	Y	Y
5.2.5.2		Y	Y	Y	Y
5.2.5.3		Y	Y	Y	Y
5.2.5.4		Y	Y	Y	Y
5.2.6	Nonconformances	-	-	-	-
5.2.6.1		Y	Y	Y	Y
5.2.6.2		Y	Y	Y	Y
5.2.7	Quality requirements and quality models	-	-	-	-
5.2.7.1		Y	Y	Y	Y
5.2.7.2		Y	Y	Y	Relevant characteristics only (e.g. suitability for safety is not relevant for cat. D software)
5.3	Risk management and critical item control	-	-	-	-
5.3.1	Risk management	Y	Y	Y	Y
5.3.2	Critical item control	-	-	-	-
5.3.2.1		Y	Y	Y	Y
5.3.2.2		Y	Y	Y	Y
5.4	Supplier selection and control	-	-	-	-
5.4.1	Supplier selection	-	-	-	-
5.4.1.1		Y	Y	Y	Expected output not required
5.4.1.2		Y	Y	Y	Y
5.4.2	Supplier requirements	-	-	-	-
5.4.2.1		Y	Y	Y	Y
5.4.2.2		Y	Y	Y	N
5.4.3	Supplier monitoring	-	-	-	-
5.4.3.1		Y	Y	Y	Y

Clause	Description	A	B	C	D
5.4.3.2		Y	Y	Y	Y
5.4.3.3		Y	Y	Y	Y
5.4.3.4		Y	Y	Y	N
5.4.4	Criticality classification	Y	Y	Y	Y
5.5	Procurement	-	-	-	-
5.5.1	Procurement documents	Y	Y	Y	Y
5.5.2	Review of procured software component list	Y	Y	Y	Y
5.5.3	Procurement details	Y	Y	Y	Y
5.5.4	Identification	Y	Y	Y	Y
5.5.5	Inspection	Y	Y	Y	Y
5.5.6	Exportability	Y	Y	Y	Y
5.6	Tools and supporting environment	-	-	-	-
5.6.1	Methods and tools	-	-	-	
5.6.1.1		Y	Y	Y	The proposed methods and tools shall have been successfully used at least in one project before (possibly a non-space project)
5.6.1.2		Y	Y	Y	Expected output not required
5.6.1.3		Y	Y	Y	Expected output not required
5.6.2	Development environment selection	-	-	-	-
5.6.2.1		Y	Y	Y	Expected output not required
5.6.2.2		Y	Y	Y	Expected output not required
5.6.2.3		Y	Y	Y	Y
5.7	Assessment and improvement process	-	-	-	-
5.7.1	Process assessment	Y	Y	Y	N
5.7.2	Assessment process	-	-	-	-
5.7.2.1		Y	Y	Y	N
5.7.2.2		Y	Y	Y	N
5.7.2.3		Y	Y	Y	N
5.7.2.4		Y	Y	Y	N
5.7.3	Process improvement	-	-	-	-
5.7.3.1		Y	Y	Y	N

Clause	Description	A	B	C	D
5.7.3.2		Y	Y	Y	N
5.7.3.3		Y	Y	Y	N
6	**Software process assurance**	-	-	-	-
6.1	Software development life cycle	-	-	-	-
6.1.1	Life cycle definition	Y	Y	Y	Y
6.1.2	Quality objectives	Y	Y	Y	Y
6.1.3	Life cycle definition review	Y	Y	Y	Y
6.1.4	Life cycle resources	Y	Y	Y	Y
6.1.5	Software validation process schedule	Y	Y	Y	Y
6.2	Requirements applicable to all software engineering processes	-	-	-	-
6.2.1	Documentation of processes	-	-	-	-
6.2.1.1		Y	Y	Y	Y
6.2.1.2		Y	Y	Y	Y
6.2.1.3		Y	Y	Y	Y
6.2.1.4		Y	Y	Y	Y
6.2.1.5		Y	Y	Y	Y
6.2.1.6		Y	Y	Y	Y
6.2.1.7		Y	Y	Y	Y
6.2.1.8		Y	Y	Y	Y
6.2.1.9		Y	Y	Y	N
6.2.2	Software dependability and safety	-	-	-	-
6.2.2.1		Y	Y	Y	Y
6.2.2.2		Y	Y	Y	N
6.2.2.3		Y	Y	Y	N
6.2.2.4		Y	Y	Y	N
6.2.2.5		Y	Y	Y	N
6.2.2.6		Y	Y	Y	N
6.2.2.7		Y	Y	Y	N
6.2.2.8		Y	Y	Y	Y
6.2.2.9		Y	Y	Y	Y
6.2.2.10		Y	Y	Y	Y
6.2.3	Handling of critical software	-	-	-	-
6.2.3.2		Y	Y	Y	N
6.2.3.3		Y	Y	Y	N
6.2.3.4		Y	Y	Y	N
6.2.3.5		Y	Y	Y	N
6.2.3.6		Y	Y	Y	N

Clause	Description	A	B	C	D
6.2.3.7		Y	Y	N	N
6.2.3.8		Y	Y	Y	N
6.2.4	Software configuration management	-	-	-	-
6.2.4.1		Y	Y	Y	Y
6.2.4.2		Y	Y	Y	Y
6.2.4.3		Y	Y	Y	Y
6.2.4.4		Y	Y	Y	Y
6.2.4.5		Y	Y	Y	Y
6.2.4.6		Y	Y	Y	Y
6.2.4.7		Y	Y	Y	Y
6.2.4.8		Y	Y	Y	Y
6.2.4.9		Y	Y	Y	Y
6.2.4.10		Y	Y	Y	Y
6.2.4.11		Y	Y	Y	Y
6.2.5	Process metrics	-	-	-	-
6.2.5.1		Y	Y	Y	Y
6.2.5.2		Y	Y	Y	Y
6.2.5.3		Y	Y	Y	Y
6.2.5.4		Y	Y	Y	Limited to number of problems detected during validation
6.2.5.5		Y	Y	Y	Y
6.2.6	Verification	-	-	-	-
6.2.6.1		Y	Y	Y	Y
6.2.6.2		Y	Y	Y	Y
6.2.6.3		Y	Y	Y	Y
6.2.6.4		Y	Y	Y	Y
6.2.6.5		Y	Y	Y	N
6.2.6.6		Y	Y	Y	N
6.2.6.7		Y	Y	Y	Y
6.2.6.8		Y	Y	Y	Y
6.2.6.9		Y	Y	Y	Y
6.2.6.10		Y	Y	Y	Y
6.2.6.11		Y	Y	Y	Y
6.2.6.12		Y	Y	Y	Y
6.2.6.13		Y	Y	N	N
6.2.7	Reuse of existing software	-	-	-	-

Clause	Description	A	B	C	D
6.2.7.1		Y	Y	Y	Y
6.2.7.2		Y	Y	Y	Y
6.2.7.3		Y	Y	Y	Y
6.2.7.4		Y	Y	Y	Bullets 3, 4, 5 and 7 not applicable. Bullet 2 limited to architectural design
6.2.7.5		Y	Y	Y	Y
6.2.7.6		Y	Y	Y	Y
6.2.7.7		Y	Y	Y	Limited to the extent to ensure maintainability of the software
6.2.7.8		Y	Y	Y	Limited to the extent to ensure maintainability of the software
6.2.7.9		Y	Y	Y	Y
6.2.7.10		Y	Y	Y	Y
6.2.7.11		Y	Y	Y	Y
6.2.8	Automatic code generation	-	-	-	-
6.2.8.1		Y	Y	Y	Y
6.2.8.2		Y	Y	Y	Y
6.2.8.3		Y	Y	Y	Y
6.2.8.4		Y	Y	Y	Y
6.2.8.5		Y	Y	Y	Y
6.2.8.6		Y	Y	Y	Y
6.2.8.7		Y	Y	Y	Y
6.3	Requirements applicable to individual software engineering processes or activities	-	-	-	-
6.3.1	Software related system requirements process	-	-	-	-
6.3.1.1		Y	Y	Y	Y
6.3.1.2		Y	Y	Y	Y
6.3.1.3		Y	Y	Y	Y
6.3.2	Software requirements analysis	-	-	-	-
6.3.2.1		Y	Y	Y	Y
6.3.2.2		Y	Y	Y	Y
6.3.2.3		Y	Y	Y	Y
6.3.2.4		Y	Y	Y	Y

Clause	Description	A	B	C	D
6.3.2.5		Y	Y	Y	Y
6.3.3	Software architectural design and design of software items	-	-	-	-
6.3.3.1		Y	Y	Y	Documentation control only
6.3.3.2		Y	Y	Y	Only recommended
6.3.3.3		Y	Y	Y	N
6.3.3.4		Y	Y	Y	Only if design standards are applied (6.3.2.2)
6.3.3.5		Y	Y	Y	N
6.3.3.6		Y	Y	Y	N
6.3.3.7		Y	Y	Y	Y
6.3.4	Coding	-	-	-	-
6.3.4.1		Y	Y	Y	Y
6.3.4.2		Y	Y	Y	Y
6.3.4.3		Y	Y	Y	N
6.3.4.4		Y	Y	Y	N
6.3.4.5		Y	Y	Y	Y
6.3.4.6		Y	Y	Y	Y
6.3.4.7		Y	Y	Y	Y
6.3.4.8		Y	Y	Y	The code shall be put under configuration control at the beginning of validation testing
6.3.5	Testing and validation	-	-	-	-
6.3.5.1		Y	Y	Y	No formal unit testing and integration activity required
6.3.5.2		Y	Y	Y	No formal unit testing and integration activity required
6.3.5.3		Y	Y	Y	Test procedures and data verified by sample

Clause	Description	A	B	C	D
6.3.5.4		Y	Y	Y	Applicable to validation and acceptance tests only
6.3.5.5		Y	Y	Y	Y
6.3.5.6		Y	Y	Y	Y
6.3.5.7		Y	Y	Y	Y
6.3.5.8		Y	Y	Y	Y
6.3.5.9		Y	Y	Y	N
6.3.5.10		Y	Y	Y	N
6.3.5.11		Y	Y	Y	Y
6.3.5.12		Y	Y	Y	Y
6.3.5.13		Y	Y	Y	Y
6.3.5.14		Y	Y	Y	Applicable to validation and acceptance tests only
6.3.5.15		Y	Y	Y	Y
6.3.5.16		Y	Y	Y	Y
6.3.5.17		Y	Y	Y	Y
6.3.5.18		Y	Y	Y	Y
6.3.5.19		Y	Y	Y	N
6.3.5.20		Y	Y	Y	Y
6.3.5.21		Y	Y	Y	Y
6.3.5.22		Y	Y	Y	Y
6.3.5.23		Y	Y	Y	Y
6.3.5.24		Y	Y	Y	Y
6.3.5.25		Y	Y	Y	Y
6.3.5.26		Y	Y	Y	Y
6.3.5.27		Y	Y	Y	Y
6.3.5.28		Y	Y	N	N
6.3.5.29		Y	Y	Y	Y
6.3.5.30		Y	Y	Y	N
6.3.5.31		Y	Y	Y	N
6.3.5.32		Y	Y	Y	Y
6.3.6	Software delivery and acceptance	-	-	-	-
6.3.6.1		Y	Y	Y	Y
6.3.6.2		Y	Y	Y	Y
6.3.6.3		Y	Y	Y	Y

Clause	Description	A	B	C	D
6.3.6.4		Y	Y	Y	Y
6.3.6.5		Y	Y	Y	Y
6.3.6.6		Y	Y	Y	Y
6.3.6.7		Y	Y	Y	Y
6.3.6.8		Y	Y	Y	Y
6.3.6.9		Y	Y	Y	Y
6.3.7	Operations	-	-	-	-
6.3.7.1		Y	Y	Y	Y
6.3.7.2		Y	Y	Bullet on safety features not applicable	Bullet on safety features not applicable
6.3.7.3		Y	Y	Y	Y
6.3.8	Maintenance	-	-	-	-
6.3.8.1		Y	Y	Y	Y
6.3.8.2		Y	Y	Y	Y
6.3.8.3		Y	Y	Y	Y
6.3.8.4		Y	Y	Y	Y
6.3.8.5		Y	Y	Y	Y
6.3.8.6		Y	Y	Y	Y
6.3.8.7		Y	Y	Y	Statistical data not collected
7	**Software product quality assurance**	-	-	-	-
7.1	Product quality objectives and metrication	-	-	-	-
7.1.1	Deriving of requirements	Y	Y	Y	Y
7.1.2	Quantitative definition of quality requirements	Y	Y	Y	Y
7.1.3	Assurance activities for product quality requirements	Y	Y	Y	Y
7.1.4	Product metrics	Y	Y	Y	Bullet 4.(a) not applicable
7.1.5	Basic metrics	Y	Y	Y	Design-relevant and fault density/failure intensity metrics not required
7.1.6	Reporting of metrics	Y	Y	Y	Y
7.1.7	Numerical accuracy	Y	Y	Y	Y
7.1.8	Analysis of software maturity	Y	Y	Y	N
7.2	Product quality requirements	-	-	-	-

Clause	Description	A	B	C	D
7.2.1	Requirements baseline and technical specification	-	-	-	-
7.2.1.1		Y	Y	Y	Y
7.2.1.2		Y	Y	Y	Y
7.2.1.3		Y	Y	Y	Y
7.2.2	Design and related documentation	-	-	-	-
7.2.2.1		Y	Y	Y	Y
7.2.2.2		Y	Y	Y	Y
7.2.2.3		Y	Y	Y	Y
7.2.3	Test and validation documentation	-	-	-	-
7.2.3.1		Y	Y	Y	Y
7.2.3.2		Y	Y	Y	Y
7.2.3.3		Y	Y	Y	Y
7.2.3.4		Y	Y	Y	Y
7.2.3.5		Y	Y	Y	Y
7.2.3.6		Y	Y	Y	Y
7.3	Software intended for reuse	-	-	-	-
7.3.1	Customer requirements	Y	Y	Y	Y
7.3.2	Separate documentation	Y	Y	Y	Y
7.3.3	Self-contained information	Y	Y	Y	Y
7.3.4	Requirements for intended reuse	Y	Y	Y	Y
7.3.5	Configuration management for intended reuse	Y	Y	Y	Y
7.3.6	Testing on different platforms	Y	Y	Y	Y
7.3.7	Certificate of conformance	Y	Y	Y	Y
7.4	Standard hardware for operational system	-	-	-	-
7.4.1	Hardware procurement	Y	Y	Y	Y
7.4.2	Service procurement	Y	Y	Y	Y
7.4.3	Constraints	Y	Y	Y	Y
7.4.4	Selection	Y	Y	Y	Y
7.4.5	Maintenance	Y	Y	Y	Y
7.5	Firmware	-	-	-	-
7.5.1	Device programming	Y	Y	Y	Y
7.5.2	Marking	Y	Y	Y	Y
7.5.3	Calibration	Y	Y	Y	Y

Annex E (informative)
List of requirements with built-in tailoring capability

The following requirements are applicable under specific conditions, as described in the requirement's text.

5.1.4.2	The software product assurance *manager*/*engineer* shall report to the project manager (through the project product assurance manager, *if any*)
5.2.2.1	The supplier shall report on a regular basis on the status of the software product assurance programme implementation, *if appropriate* as part of the overall product assurance reporting of the project.
6.2.3.4	*In case* of minor changes in tools that affect the generation of the executable code, a binary comparison of the executable code generated by the different tools can be used to verify that no modifications are introduced
6.2.6.13	This requirement is applicable *where* the risks associated with the project justify the costs involved. The customer *may* consider a less rigorous level of independence, e.g. an independent team in the same organization.

The following requirements foresee an agreement between the customer and the supplier.

6.3.2.5	Prior to the technical specification elaboration, customer and supplier shall agree on the following principles and rules as a minimum: […].
6.3.5.2	Based on the criticality of the software, test coverage goals for each testing level shall be agreed between the customer and the supplier and their achievement monitored by metrics: […].

Annex F (informative) Document organization and content at each milestone

F.1 Introduction

The following table shows the organization of the Expected Output of the clauses of this Standard, sorted per review, then per destination file, then per DRD.

When no DRD is available, "-" is shown.

F.2 ECSS-Q-ST-80 Expected Output at SRR

Clause	Expected Output	Dest. File	DRD	Section
7.1.1.a	Requirement baseline	RB	SSS	<5.9>
7.1.2.a	Requirement baseline	RB	SSS	<5.9>
7.2.1.1.a	Requirement baseline	RB	SSS	<5.9>
7.2.1.3.a	Requirement baseline	RB	SSS	<5.1>
5.4.4	Safety and dependability analyses results for lower level suppliers	RB	-	
5.1.2.1	Software product assurance plan	PAF	SPAP	<5.1>
5.1.2.2	Software product assurance plan	PAF	SPAP	<5.1>, <5.2>
5.1.2.3	Software product assurance plan	PAF	SPAP	<5.1>
5.1.3.1	Software product assurance plan	PAF	SPAP	<5.3>
5.1.3.2	Software product assurance plan	PAF	SPAP	<5.1>, <5.3>
5.1.4.1	Software product assurance plan	PAF	SPAP	<5.1>, <5.3>
5.2.1.1	Software product assurance plan	PAF	SPAP	All
5.2.1.5	Software product assurance plan	PAF	SPAP	<8>
5.2.6.1.a.a	NCR SW procedure as part of the Software product assurance plan	PAF	SPAP	
5.2.6.2.	Software product assurance plan	PAF	SPAP	<6.4>
5.6.1.1	Software product assurance plan	PAF	SPAP	<5.8>
6.1.1	Software product assurance plan	PAF	SPAP	<6.1>

Clause	Expected Output	Dest. File	DRD	Section
6.1.5	Software product assurance plan	PAF	SPAP	<6.1>
6.2.1.4	Software product assurance plan	PAF	SPAP	<6.2>
6.2.4.8.a	Software product assurance plan	PAF	SPAP	<6.4>
6.2.4.9	Software product assurance plan	PAF	SPAP	<6.4>
6.2.4.11.a	Software product assurance plan	PAF	SPAP	<6.4>
6.2.5.1	Software product assurance plan	PAF	SPAP	<6.5>
6.2.5.2	Software product assurance plan	PAF	SPAP	<6.5>
6.2.7.2.a	Software reuse approach, including approach to delta qualification	PAF	SPAP	<6.6>
6.2.7.3.a	Software reuse approach, including approach to delta qualification	PAF	SPAP	<6.6>
6.2.7.4.a	Software reuse approach, including approach to delta qualification	PAF	SPAP	<6.6>
6.2.7.5.a	Software reuse approach, including approach to delta qualification	PAF	SPAP	<6.6>
7.1.3	Software product assurance plan	PAF	SPAP	<7>
7.1.4	Software product assurance plan	PAF	SPAP	<7>
7.1.5	Software product assurance plan	PAF	SPAP	<7>
7.2.2.3.a	Software product assurance plan	PAF	SPAP	<6.7>
5.2.2.3	Software product assurance milestone report	PAF	SPAMR	All
5.6.1.2	Software product assurance milestone report	PAF	SPAMR	<5>
6.2.6.12	Software product assurance milestone report	PAF	SPAMR	<4>
5.2.3	Audit plan and schedule	PAF	-	
5.4.2.1	Software product assurance requirements for suppliers	PAF	-	
5.4.2.2	Software product assurance requirements for suppliers	PAF	-	
6.2.2.1	Criticality classification of software products	PAF	-	
6.2.8.4	Modelling standards	PAF	-	
6.3.3.2	Design standards	PAF	-	
7.4.1.b	Receiving inspection report	PAF	-	
5.5.2	Software development plan	MGT	SDP	<4.8>
5.6.2.1	Software development plan	MGT	SDP	<5.4>
5.6.2.2	Software development plan	MGT	SDP	<5.4>
6.2.4.2	Software configuration management plan	MGT	SCMP	

Clause	Expected Output	Dest. File	DRD	Section
7.3.5	Configuration management for reusable components	MGT	SCMP	
5.1.5.1	Training plan	MGT	-	
5.2.6.1.b	Identification of SW experts in NRB	MGT	-	
5.5.3	Procurement data	MGT	-	
6.2.7.2.b	Software reuse file	DJF	SRF	<6>
6.2.7.3.b	Software reuse file	DJF	SRF	<4>, <5>
6.2.7.4.b	Software reuse file	DJF	SRF	<5>
6.2.7.5.b	Software reuse file	DJF	SRF	<6>
6.2.7.6	Software reuse file	DJF	SRF	<4>, <5>
6.2.7.7	Software reuse file	DJF	SRF	<8>
6.2.7.8	Software reuse file	DJF	SRF	<8>
6.2.7.11	Software reuse file	DJF	SRF	<9>
6.2.6.4	Software problem reports	DJF	-	
6.2.6.13.a	ISVV plan	DJF	-	
6.3.5.8	Software problem reports	DJF	-	
6.3.5.28.a	ISVV plan	DJF	-	
7.4.1.a	Justification of selection of operational ground equipment	DJF	-	
7.4.2	Justification of selection of operational support services	DJF	-	
7.4.3	Justification of selection of operational ground equipment	DJF	-	
7.4.4	Justification of selection of operational ground equipment	DJF	-	

F.3 ECSS-Q-ST-80 Expected Output at PDR

Clause	Expected output	Dest. File	DRD	Section
6.3.2.4	Software requirements specification	TS	SRS	<5>
7.1.1.b	Technical specification	TS	SRS	<5.10>
7.1.2.b	Technical specification	TS	SRS	<5.10>
7.2.1.1.b	Technical specification	TS	SRS	<5.10>
7.2.1.3.b	Technical specification	TS	SRS	<6>
7.3.4	Technical specification for reusable components	TS	-	
5.2.1.1	Software product assurance	PAF	SPAP	All

Clause	Expected output	Dest. File	DRD	Section
	plan			
5.2.1.5	Software product assurance plan	PAF	SPAP	<8>
5.2.6.2.	Software product assurance plan	PAF	SPAP	<6.4>
5.2.7.1	Software product assurance plan	PAF	SPAP	<5.5>
5.2.7.2	Software product assurance plan	PAF	SPAP	<5.5>
5.4.3.3	Next level suppliers' software product assurance plan	PAF	SPAP	All
5.4.3.4	Next level suppliers' software product assurance plan	PAF	SPAP	All
5.6.1.1	Software product assurance plan	PAF	SPAP	<5.8>
6.1.1	Software product assurance plan	PAF	SPAP	<6.1>
6.1.5	Software product assurance plan	PAF	SPAP	<6.1>
6.2.1.4	Software product assurance plan	PAF	SPAP	<6.2>
6.2.2.5.a	Software product assurance plan	PAF	SPAP	<6.3>
6.2.3.2	Software product assurance plan	PAF	SPAP	<6.3>
6.2.3.4	Software product assurance plan	PAF	SPAP	<6.7>
6.2.3.5	Software product assurance plan	PAF	SPAP	<6.7>
6.2.4.8.a	Software product assurance plan	PAF	SPAP	<6.4>
6.2.4.9	Software product assurance plan	PAF	SPAP	<6.4>
6.2.4.11.a	Software product assurance plan	PAF	SPAP	<6.4>
6.2.5.1	Software product assurance plan	PAF	SPAP	<6.5>
6.2.5.2	Software product assurance plan	PAF	SPAP	<6.5>
6.2.7.2.a	Software reuse approach, including approach to delta	PAF	SPAP	<6.6>

Clause	Expected output	Dest. File	DRD	Section
	qualification			
6.2.7.3.a	Software reuse approach, including approach to delta qualification	PAF	SPAP	<6.6>
6.2.7.4.a	Software reuse approach, including approach to delta qualification	PAF	SPAP	<6.6>
6.2.7.5.a	Software reuse approach, including approach to delta qualification	PAF	SPAP	<6.6>
6.3.3.3	Software product assurance plan	PAF	SPAP	<6.8>
6.3.3.5	Software product assurance plan	PAF	SPAP	<6.7>
6.3.3.7.a	Software product assurance plan	PAF	SPAP	<6.7>
6.3.4.3	Software product assurance plan	PAF	SPAP	<6.8>
6.3.4.6	Software product assurance plan	PAF	SPAP	<6.8>
6.3.5.1	Software product assurance plan	PAF	SPAP	<6.7>
6.3.5.2	Software product assurance plan	PAF	SPAP	<6.7>
7.1.3	Software product assurance plan	PAF	SPAP	<7>
7.1.4	Software product assurance plan	PAF	SPAP	<7>
7.1.5	Software product assurance plan	PAF	SPAP	<7>
7.2.2.3.a	Software product assurance plan	PAF	SPAP	<6.7>
7.5.1	Software product assurance plan	PAF	SPAP	<6.8>
7.5.2	Software product assurance plan	PAF	SPAP	<6.8>
5.2.2.3	Software product assurance milestone report	PAF	SPAMR	All
5.6.1.2	Software product assurance milestone report	PAF	SPAMR	<5>
6.2.3.3	Software product assurance milestone report	PAF	SPAMR	<4>
6.2.6.12	Software product assurance	PAF	SPAMR	<4>

Clause	Expected output	Dest. File	DRD	Section
	milestone report			
5.2.5.1	Software problem reporting procedures	PAF	-	
5.2.5.2	Software problem reporting procedures	PAF	-	
5.2.5.3	Software problem reporting procedures	PAF	-	
5.5.5	Receiving inspection report	PAF	-	
6.2.1.6	Procedures and standards	PAF	-	
6.2.1.7	Procedures and standards	PAF	-	
6.2.2.1	Criticality classification of software products	PAF	-	
6.2.2.2	Software dependability and safety analysis report	PAF	-	
6.2.2.3	Criticality classification of software components	PAF	-	
6.2.2.7	Software dependability and safety analysis report	PAF	-	
6.2.2.10.b	Software dependability and safety analysis report	PAF	-	
6.2.8.4	Modelling standards	PAF	-	
6.3.3.2	Design standards	PAF	-	
6.3.4.1	Coding standards	PAF	-	
6.3.4.2	Coding standards	PAF	-	
6.3.4.4	Coding standards and description of tools	PAF	-	
7.4.1.b	Receiving inspection report	PAF	-	
5.5.2	Software development plan	MGT	SDP	<4.8>
5.6.2.1	Software development plan	MGT	SDP	<5.4>
5.6.2.2	Software development plan	MGT	SDP	<5.4>
6.3.4.5	Software development plan	MGT	SDP	<5.4>
6.2.4.2	Software configuration management plan	MGT	SCMP	
7.3.5	Configuration management for reusable components	MGT	SCMP	
5.5.3	Procurement data	MGT	-	
7.1.7	Numerical accuracy analysis	DJF	SVR	<6>
6.2.6.1	Software verification plan	DJF	SVerP	<6.3>
6.2.8.2	Validation and testing documentation	DJF	SValP	<4.1>

Clause	Expected output	Dest. File	DRD	Section
6.2.8.7	Validation and testing documentation	DJF	SValP	<4.1>
6.3.5.22	Test and validation documentation	DJF	SValP	<4>
6.3.5.23	Test and validation documentation	DJF	SValP	<4.4>
6.3.5.24	Test and validation documentation	DJF	SValP	<4.6>
6.3.5.25	Test and validation documentation	DJF	SValP	<5>
6.3.5.29	Test and validation documentation	DJF	SValP	<6>
6.2.8.2	Validation and testing documentation	DJF	SUITP	<7.6>
6.2.8.7	Validation and testing documentation	DJF	SUITP	<7.6>
6.3.5.22	Test and validation documentation	DJF	SUITP	<5>
6.3.5.23	Test and validation documentation	DJF	SUITP	<5.3>
6.3.5.24	Test and validation documentation	DJF	SUITP	<5.5>
6.3.5.25	Test and validation documentation	DJF	SUITP	<9.2>, <10>
6.2.7.2.b	Software reuse file	DJF	SRF	<6>
6.2.7.3.b	Software reuse file	DJF	SRF	<4>, <5>
6.2.7.4.b	Software reuse file	DJF	SRF	<5>
6.2.7.5.b	Software reuse file	DJF	SRF	<6>
6.2.7.6	Software reuse file	DJF	SRF	<4>, <5>
6.2.7.7	Software reuse file	DJF	SRF	<8>
6.2.7.8	Software reuse file	DJF	SRF	<8>
6.2.7.11	Software reuse file	DJF	SRF	<9>
6.2.6.4	Software problem reports	DJF	-	
6.2.6.13.a	ISVV plan	DJF	-	
6.2.6.13.a	ISVV report	DJF	-	
6.3.5.8	Software problem reports	DJF	-	
6.3.5.28.a	ISVV plan	DJF	-	
6.3.5.28.b	ISVV report	DJF	-	
7.4.1.a	Justification of selection of operational ground equipment	DJF	-	

Clause	Expected output	Dest. File	DRD	Section
7.4.2	Justification of selection of operational support services	DJF	-	
7.4.3	Justification of selection of operational ground equipment	DJF	-	
7.4.4	Justification of selection of operational ground equipment	DJF	-	
7.2.2.3.b	Justification of design choices	DDF	SDD	<4.5>

F.4 ECSS-Q-ST-80 Expected Output at CDR

Clause	Expected output	Dest. File	DRD	Section
5.2.1.3	Software product assurance plan	PAF	SPAP	All
6.2.2.5.a	Software product assurance plan	PAF	SPAP	<6.3>
6.2.3.2	Software product assurance plan	PAF	SPAP	<6.3>
6.2.3.4	Software product assurance plan	PAF	SPAP	<6.7>
6.2.3.5	Software product assurance plan	PAF	SPAP	<6.7>
5.2.2.3	Software product assurance milestone report	PAF	SPAMR	All
6.2.3.3	Software product assurance milestone report	PAF	SPAMR	<4>
6.2.6.12	Software product assurance milestone report	PAF	SPAMR	<4>
5.5.5	Receiving inspection report	PAF	-	
6.2.2.5	Software dependability and safety analysis report	PAF	-	
6.2.2.6	Software dependability and safety analysis report	PAF	-	
6.2.2.7	Software dependability and safety analysis report	PAF	-	
6.2.2.10.b	Software dependability and safety analysis report	PAF	-	
6.3.5.7	Statement of compliance with test plans and procedures	PAF	-	
6.3.5.11	Statement of compliance with test plans and procedures	PAF	-	
6.2.8.2	Validation and testing documentation	DJF	SVS	<5.6>

Clause	Expected output	Dest. File	DRD	Section
6.2.8.7	Validation and testing documentation	DJF	SVS	<5.6>
6.3.5.25	Test and validation documentation	DJF	SVS	<7.2>, <8>
6.3.5.29	Test and validation documentation	DJF	SVS	<6>
6.3.5.32	Software validation specification	DJF	SVS	<5>
6.2.6.5	Software verification report	DJF	SVR	<4.4>
6.2.6.6	Software verification report	DJF	SVR	<4.4>
7.1.7	Numerical accuracy analysis	DJF	SVR	<6>
7.2.3.6	Software verification report	DJF	SVR	<4.5>
6.2.8.2	Validation and testing documentation	DJF	SUITP	<7.6>
6.2.8.7	Validation and testing documentation	DJF	SUITP	<7.6>
6.3.5.22	Test and validation documentation	DJF	SUITP	<5>
6.3.5.23	Test and validation documentation	DJF	SUITP	<5.3>
6.3.5.24	Test and validation documentation	DJF	SUITP	<5.5>
6.3.5.25	Test and validation documentation	DJF	SUITP	<9.2>, <10>
6.2.7.9	Software reuse file	DJF	SRF	<8>
6.2.7.11	Software reuse file	DJF	SRF	<9>
6.2.6.4	Software problem reports	DJF	-	
6.2.6.13.a	ISVV report	DJF	-	
6.3.5.6	Nonconformance reports and software problem reports	DJF	-	
6.3.5.8	Software problem reports	DJF	-	
6.3.5.13	Testing and validation reports	DJF	-	
6.3.5.16	Updated test documentation	DJF	-	
6.3.5.17	Updated test documentation	DJF	-	
6.3.5.18	Updated test documentation	DJF	-	
6.3.5.28.b	ISVV report	DJF	-	
6.3.5.30	Testing and validation reports	DJF	-	
6.3.5.31	Testing and validation reports	DJF	-	
7.3.6	Verification and validation documentation for reusable components	DJF	-	

Clause	Expected output	Dest. File	DRD	Section
7.3.7	Verification and validation documentation for reusable components	DJF	-	
6.2.4.4	Software configuration file	DDF	SCF	All
6.2.4.5	Software configuration file	DDF	SCF	All
6.2.4.8.b	Software configuration file	DDF	SCF	All
7.2.2.3.b	Justification of design choices	DDF	SDD	<4.5>

F.5 ECSS-Q-ST-80 Expected Output at QR

Clause	Expected output	Dest. File	DRD	Section
5.2.1.3	Software product assurance plan	PAF	SPAP	All
5.2.2.3	Software product assurance milestone report	PAF	SPAMR	All
6.2.3.3	Software product assurance milestone report	PAF	SPAMR	<4>
6.2.6.12	Software product assurance milestone report	PAF	SPAMR	<4>
5.5.5	Receiving inspection report	PAF	-	
6.2.2.5	Software dependability and safety analysis report	PAF	-	
6.2.2.6	Software dependability and safety analysis report	PAF	-	
6.2.2.10.b	Software dependability and safety analysis report	PAF	-	
6.3.5.7	Statement of compliance with test plans and procedures	PAF	-	
6.3.5.11	Statement of compliance with test plans and procedures	PAF	-	
6.3.8.1	Maintenance plan	MF	-	
6.3.8.2	Maintenance plan	MF	-	
6.3.8.4	Maintenance plan	MF	-	
6.3.8.5	Maintenance plan	MF	-	
6.2.8.2	Validation and testing documentation	DJF	SVS	<5.6>
6.2.8.7	Validation and testing documentation	DJF	SVS	<5.6>
6.3.5.25	Test and validation documentation	DJF	SVS	<7.2>, <8>
6.3.5.29	Test and validation	DJF	SVS	<6>

Clause	Expected output	Dest. File	DRD	Section
	documentation			
6.3.5.32	Software validation specification	DJF	SVS	<5>
6.2.6.5	Software verification report	DJF	SVR	<4.4>
6.2.6.6	Software verification report	DJF	SVR	<4.4>
7.1.7	Numerical accuracy analysis	DJF	SVR	<6>
7.2.3.6	Software verification report	DJF	SVR	<4.5>
6.2.7.9	Software reuse file	DJF	SRF	<8>
6.2.7.11	Software reuse file	DJF	SRF	<9>
6.2.6.4	Software problem reports	DJF	-	
6.2.6.13.a	ISVV report	DJF	-	
6.3.5.6	Nonconformance reports and software problem reports	DJF	-	
6.3.5.8	Software problem reports	DJF	-	
6.3.5.13	Testing and validation reports	DJF	-	
6.3.5.16	Updated test documentation	DJF	-	
6.3.5.17	Updated test documentation	DJF	-	
6.3.5.18	Updated test documentation	DJF	-	
6.3.5.28.b	ISVV report	DJF	-	
6.3.5.30	Testing and validation reports	DJF	-	
6.3.5.31	Testing and validation reports	DJF	-	
6.2.4.4	Software configuration file	DDF	SCF	All
6.2.4.5	Software configuration file	DDF	SCF	All
6.2.4.8.b	Software configuration file	DDF	SCF	All

F.6 ECSS-Q-ST-80 Expected Output at AR

Clause	Expected output	Dest. File	DRD	Section
5.2.1.3	Software product assurance plan	PAF	SPAP	All
5.2.1.4	Software product assurance plan	PAF	SPAP	<5.10>
5.2.2.3	Software product assurance milestone report	PAF	SPAMR	All
6.2.3.3	Software product assurance milestone report	PAF	SPAMR	<4>
6.2.6.12	Software product assurance milestone report	PAF	SPAMR	<4>
6.2.2.5	Software dependability and safety analysis report	PAF	-	
6.2.2.6	Software dependability and safety	PAF	-	

Clause	Expected output	Dest. File	DRD	Section
	analysis report			
6.2.2.10	Software dependability and safety analysis report	PAF	-	
6.3.5.7	Statement of compliance with test plans and procedures	PAF	-	
6.3.5.11	Statement of compliance with test plans and procedures	PAF	-	
6.3.8.1	Maintenance plan	MF	-	
6.3.8.2	Maintenance plan	MF	-	
6.3.8.4	Maintenance plan	MF	-	
6.3.8.5	Maintenance plan	MF	-	
6.2.8.2	Validation and testing documentation	DJF	SVS	<5.6>
6.2.8.7	Validation and testing documentation	DJF	SVS	<5.6>
6.3.5.25	Test and validation documentation	DJF	SVS	<7.2>, <8>
6.3.5.29	Test and validation documentation	DJF	SVS	<6>
6.3.5.32	Software validation specification	DJF	SVS	<5>
6.2.6.5	Software verification report	DJF	SVR	<4.4>
6.2.6.6	Software verification report	DJF	SVR	<4.4>
7.2.3.6	Software verification report	DJF	SVR	<4.5>
6.2.7.9	Software reuse file	DJF	SRF	<8>
6.2.7.11	Software reuse file	DJF	SRF	<9>
6.2.6.4	Software problem reports	DJF	-	
6.2.6.13.a	ISVV report	DJF	-	
6.3.5.6	Nonconformance reports and software problem reports	DJF	-	
6.3.5.8	Software problem reports	DJF	-	
6.3.5.13	Testing and validation reports	DJF	-	
6.3.5.16	Updated test documentation	DJF	-	
6.3.5.17	Updated test documentation	DJF	-	
6.3.5.18	Updated test documentation	DJF	-	
6.3.5.27	Test and validation documentation	DJF	-	
6.3.5.28.b	ISVV report	DJF	-	
6.3.5.30	Testing and validation reports	DJF	-	
6.3.5.31	Testing and validation reports	DJF	-	
6.3.6.3	Acceptance test plan	DJF	-	

Clause	Expected output	Dest. File	DRD	Section
6.3.6.7	Nonconformance reports	DJF	-	
6.3.6.8	Acceptance test report	DJF	-	
6.3.6.9	Acceptance test report	DJF	-	
6.2.4.4	Software configuration file	DDF	SCF	All
6.2.4.5	Software configuration file	DDF	SCF	All
6.2.4.8.b	Software configuration file	DDF	SCF	All
6.3.6.1	Installation procedure	DDF	SCF	<4.2>

F.7 ECSS-Q-ST-80 Expected Output not associated with any specific milestone review

Clause	Expected output	Dest. File	DRD	Section
5.1.5.2	Records of training and experience	PAF	-	
5.2.2.1	Software product assurance report	PAF	-	
5.2.2.2	Software product assurance report	PAF	-	
5.2.4.a	Preliminary alert information	PAF	-	
5.2.4.b	Alert information	PAF	-	
5.4.1.1.a	Results of pre-award audits and assessments	PAF	-	
5.4.1.1.b	Records of procurement sources	PAF	-	
5.6.1.3	Software product assurance reports	PAF	-	
5.7.1	Software process assessment records: Overall assessments and improvement programme plan	PAF	-	
5.7.2.1.a	Software process assessment record: assessment model	PAF	-	
5.7.2.1.b	Software process assessment record: assessment method	PAF	-	
5.7.2.2.a	Software process assessment record: evidence of conformance of the process assessment model	PAF	-	
5.7.2.2.b	Software process assessment record: assessment method	PAF	-	
5.7.2.3	Software process assessment record: Software process assessment recognition evidence	PAF	-	
5.7.2.4	Software process assessment record: competent assessor justification	PAF	-	
5.7.3.1	Software process assessment	PAF	-	

Clause	Expected output	Dest. File	DRD	Section
	records: improvement plan			
5.7.3.2	Software process assessment records: improvement process	PAF	-	
5.7.3.3	Software process assessment records: evidence of improvements	PAF	-	
6.2.5.4	Software product assurance reports	PAF	-	
6.2.5.5	Software product assurance reports	PAF	-	
6.2.6.2	Software product assurance reports	PAF	-	
6.2.6.3	Software product assurance reports	PAF	-	
6.2.6.7	Software product assurance reports	PAF	-	
6.2.6.9	Review and inspection plans or procedures	PAF	-	
6.2.6.10	Review and inspection plans or procedures	PAF	-	
6.2.6.11	Review and inspection records	PAF	-	
6.2.8.5	Software product assurance reports	PAF	-	
6.3.3.4	Software product assurance reports	PAF	-	
6.3.3.6	Software product assurance reports	PAF	-	
6.3.3.7.b	Software product assurance reports	PAF	-	
6.3.4.7	Software product assurance reports	PAF	-	
6.3.5.3	Software product assurance reports	PAF	-	
6.3.5.5	Software product assurance reports	PAF	-	
6.3.5.12	Software product assurance reports	PAF	-	
7.1.6	Software product assurance reports	PAF	-	
7.1.7	Software product assurance reports	PAF	-	
6.3.8.6	Maintenance records	MF	-	
6.3.8.7	Maintenance records	MF	-	
5.2.6.1.a.b	Nonconformance reports	DJF	-	
5.4.1.2	Software reuse file	DJF	SRF	All
6.2.4.3.a	Software configuration file	DDF	SCF	All
6.2.4.3.b	Software release document	DDF	SRelD	All
6.2.4.10	Software configuration file	DDF	SCF	All
6.2.4.11.b	Labels	DDF	-	

Bibliography

ECSS-S-ST-00	ECSS system — Description, implementation and general requirement
ECSS-Q-HB-80-02	Space product assurance — Software process assessment and improvement
ECSS-Q-HB-80-03	Space product assurance — Software dependability and safety methods and techniques
ECSS-Q-HB-80-04	Space product assurance — Software metrication programme definition and implementation
ECSS-Q-ST-30-02	Space product assurance — Failure modes, effects (and criticality) analysis
IEEE 610.12:1990	IEEE Standard Glossary of software engineering terminology
IEEE 1028-1997	IEEE Standard for Software Reviews
ISO 9000:2000	Quality management systems — Fundamentals and vocabulary
ISO 9126-1:2001	Software engineering — Product quality — Part 1: Quality model
ISO/IEC 12207:1995	Information technology — Software life cycle processes
ISO/IEC 15504:1998	Information technology — Software process assessment
RTCA/DO-178B	Software considerations in airborne systems and equipment certification
CMU/SEI-92-TR-022	Software Quality Measurement: A framework for counting problems and defects
CMU/SEI-2006-TR-008	CMMI for Development, Version 1.2