



# Space engineering

---

## Software - Part 1: Principles and requirements

Published by: ESA Publications Division  
ESTEC, P.O. Box 299,  
2200 AG Noordwijk,  
The Netherlands

ISSN: 1028-396X

Price: € 30

Copyright: © 2003 by the European Space Agency for the members of ECSS

Printed in : 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.

This Standard is published in two parts:

- Part 1: defining the principles and requirements applicable to space software engineering, and
- Part 2: defining the content of the document requirements definitions (DRDs) that are specifically referenced in Part 1.

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.

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

Significant changes between this version and the previous version are:

- alignment of the standard to the ECSS-Procedure-13B rev.1,
- mapping of ECSS software processes to the ISO 12207 processes,
- renaming of most of the software development processes (e.g. the former “software requirement engineering process” has become the “software requirements and architecture process” and the former “software design engineering process” has become the “software design and implementation engineering process”). The software validation activities with respect to the technical specification have been moved to the software validation process,
- split of the former software verification and validation (supporting) process into two separate (primary) ones: the software validation process and the software verification process. The principle of invocation allowing a subclause to call the services of another one has disappeared,
- the former software validation and acceptance process has been reshaped (the software validation activities with respect to the requirements baseline moved to the software validation process) and renamed “software delivery and installation process”,

- the “detailed design review” logic has been inserted (DDR appears now on the figures of clause 4 and in the expected outputs of the relevant subcauses to be reviewed at those milestones),
- re-enforcement of the link with the ECSS-E-10 system engineering Standard,
- identification of two steps of software validation within the software validation process: one with respect to the technical specification, and one with respect to the requirements baseline,
- identification of the acceptance process as a customer activity,
- merging of all the additional specific requirements for the space segment software in the main clause 5 (Requirements),
- revision of requirements relevant to software reuse and to MMI software,
- identification of the management and maintenance documentation files,

This Standard has been prepared by the ECSS-E-40 Working Group, reviewed by the Engineering Panel and the ECSS Executive, and approved by the ECSS Steering Board.

This version B cancels and replaces ECSS-E-40A.

---

# Contents

<b>Foreword</b>	<b>3</b>
<b>1 Scope</b>	<b>7</b>
<b>2 Normative references</b>	<b>9</b>
<b>3 Terms, definitions and abbreviated terms</b>	<b>11</b>
3.1 Terms and definitions	11
3.2 Abbreviated terms	14
<b>4 Space system software engineering</b>	<b>17</b>
4.1 Introduction	17
4.2 Overview of space system software engineering processes	18
4.3 Organization of this Standard	25
4.4 Relation to other ECSS Standards	26
4.5 Tailoring of this Standard	30
<b>5 Requirements</b>	<b>31</b>
5.1 Introduction	31
5.2 System engineering processes related to software	32
5.3 Software management process	38
5.4 Software requirements and architecture engineering process	43
5.5 Software design and implementation engineering process	47
5.6 Software validation process	50
5.7 Software delivery and acceptance process	53
5.8 Software verification process	55
5.9 Software operation process	60
5.10 Software maintenance process	63

## **Annex A (normative) Software documentation ..... 69**

A.1	Introduction .....	69
A.2	Requirements baseline (RB) .....	70
A.3	Technical specification (TS) .....	72
A.4	Design definition file (DDF) .....	73
A.5	Design justification file (DJF) .....	76
A.6	Management file (MGT) .....	86
A.7	Maintenance file (MF) .....	88
A.8	Operational documentation (OP) .....	89
A.9	Product assurance file (PAF) .....	90
A.10	System level documentation .....	101

## **Annex B (informative) References to other ECSS Standards ..... 103**

## **Annex C (informative) Tailoring guidelines ..... 105**

C.1	Introduction .....	105
C.2	How to tailor .....	105
C.3	Who tailors? .....	107

## **Bibliography ..... 109**

### **Figures**

Figure 1: Software life cycle processes in ECSS Standards .....	18
Figure 2: The recursive customer - supplier model .....	19
Figure 3: Overview of the software life cycle processes .....	21
Figure 4: Process constraints .....	22
Figure 5: Accommodation of different software life cycles .....	22
Figure 6: Structure of this Standard .....	26
Figure 7: ECSS-E-10 and ECSS-E-40 relationship .....	29
Figure A-1: Overview of software documents .....	69

## Scope

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

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

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

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

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

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

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

The scope of this Standard is the software developed as part of a space project, i.e. “Space system product software”. This Standard also applies to the development of non-deliverable software that affects the quality of the deliverable product. It is not intended to cover software developments beyond the scope of the ECSS System of Standards. An example is the development of commercial software packages, where software is developed for a (large) volume market and not just for a single customer, and the main requirement analysis consists of market analysis, combined with a marketing strategy.

When viewed in a specific project context, the requirements defined in this Standard should be tailored to match the requirements of a particular profile and circumstances of a project (see annex C).

NOTE Tailoring is a process by which individual requirements or specifications, standards and related documents are evaluated and made applicable to a specific project, by selection and in some exceptional cases, modification of existing or addition of new requirements.  
[ECSS-M-00-02A, clause 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 revisions 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 most recent editions of the normative documents indicated below. For undated references, the latest edition of the publication referred to applies.

ECSS-P-001	Glossary of terms
ECSS-E-10	Space engineering — System engineering
ECSS-M-00	Space project management — Policy and principles
ECSS-M-10	Space project management — Project breakdown structures
ECSS-M-20	Space project management — Project organization
ECSS-M-30	Space project management — Project phasing and planning
ECSS-M-40	Space project management — Configuration management
ECSS-Q-20	Space product assurance — Quality assurance
ECSS-Q-80B	Space product assurance — Software product assurance

*(This page is intentionally left blank)*

## Terms, definitions and abbreviated terms

### 3.1 Terms and definitions

The following terms and definitions are specific to this Standard in the sense that they are complementary or additional with respect to those contained in ECSS-P-001.

#### 3.1.1

##### **acceptance test**

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

[adapted from ISO/IEC 2382-20:1990]

#### 3.1.2

##### **configurable code**

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

#### 3.1.3

##### **critical software**

software supporting a safety or dependability critical function that if incorrect or inadvertently executed can result in catastrophic or critical consequences

NOTE For the definition of catastrophic and critical see ECSS-Q-30 and ECSS-Q-40.

#### 3.1.4

##### **deactivated code**

code that, although incorporated through correct design and coding, is not intended to execute in any software product configuration

#### 3.1.5

##### **logical model**

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

### **3.1.6**

#### **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]

### **3.1.7**

#### **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

### **3.1.8**

#### **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]

### **3.1.9**

#### **migration**

porting of a software product to a new environment

### **3.1.10**

#### **portability (a quality characteristic)**

capability of software to be transferred from one environment to another

### **3.1.11**

#### **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 subcharacteristics.

### **3.1.12**

#### **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]

### **3.1.13**

#### **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]

### **3.1.14**

#### **reusability**

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

[IEEE 610.12:1990]

### **3.1.15**

#### **singular input**

individual parameter stress testing

**3.1.16****software**

see 3.1.21 “software product”

**3.1.17****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 modules, 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.

**3.1.18****software item**

see 3.1.21 “software product”

**3.1.19****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.

**3.1.20****software observability**

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

**3.1.21****software product**

set of computer programs, procedures, documentation and their associated data

**3.1.22****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

**3.1.23****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.

**3.1.24****state (of the software)**

level of maturity of the software product along the life cycle

NOTE It can have four different statuses, “specified”, “defined”, “qualified” or “accepted”, depending on the success in achieving the PDR, CDR, QR or AR milestones.

#### **3.1.25**

##### **stress test**

test that evaluates a system or software component at or beyond its value limits

#### **3.1.26**

##### **unit test**

test of software unit to ensure that there are no programming errors

#### **3.1.27**

##### **unreachable code**

code that cannot be executed due to design or coding error

#### **3.1.28**

##### **usability (a quality characteristic)**

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

#### **3.1.29**

##### **validation**

confirmation, through the provision of objective evidence, that the requirements for a specific intended use or application have been fulfilled

[ISO 9000:2000]

NOTE The validation process (for software) is the process to confirm that the requirements baseline functions and performances are correctly and completely implemented in the final product.

#### **3.1.30**

##### **verification**

confirmation, through the provision of objective evidence, that specified requirements have been fulfilled

[ISO 9000:2000]

NOTE The verification process (for software) is the 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 Abbreviated terms**

The following abbreviated terms are defined and used within this Standard:

<b>Abbreviation</b>	<b>Meaning</b>
<b>AR</b>	acceptance review
	NOTE The term SW-AR is often used for clarity to denote ARs that solely involve software products.
<b>CDR</b>	critical design review
	NOTE The term SW-CDR is often used for clarity to denote CDRs that solely involve software products.
<b>COTS</b>	commercial off-the-shelf
<b>CPU</b>	central processing unit
<b>DDF</b>	design definition file

<b>DDR</b>	detailed design review
<b>DJF</b>	design justification file
<b>ECSS</b>	European Cooperation for Space Standardization
<b>HSIA</b>	hardware-software interaction analysis
<b>HW</b>	hardware
<b>ICD</b>	interface control document
<b>IRD</b>	interface requirements document
<b>ISO</b>	International Organization for Standardization
<b>ISV</b>	independent software validation
<b>ISVV</b>	independent software verification and validation
<b>MGT</b>	management file
<b>MF</b>	maintenance file
<b>MMI</b>	man-machine interface
<b>MOTS</b>	modified off-the-shelf
<b>OP</b>	operational plan
<b>ORR</b>	operational readiness review
<b>PDR</b>	preliminary design review
NOTE	The term SW-PDR is often used for clarity to denote PDRs that solely involve software products.
<b>QR</b>	qualification review
NOTE	The term SW-QR is often used for clarity to denote QRs that solely involve software products.
<b>RB</b>	requirements baseline
<b>SDE</b>	software development environment
<b>SPA</b>	software product assurance
<b>SPR</b>	software problem report
<b>SRB</b>	software review board
<b>SRR</b>	system requirements review
NOTE	The term SW-SRR is often used for clarity to denote SRRs that solely involve software products.
<b>SW</b>	software
<b>SWE</b>	software engineering
<b>TS</b>	technical specification
<b>V&amp;V</b>	verification and validation

*(This page is intentionally left blank)*



## Space system software engineering

### 4.1 Introduction

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

The context of space software engineering is the overall space system engineering process. This clause 4 defines the general relationships between the software engineering processes and the general engineering processes of space systems.

The software engineering discipline differs from the other engineering disciplines covered by ECSS in one important aspect: software does not in itself produce heat, have mass or any other physical characteristics. The software engineering is a purely intellectual activity and a principle output of the activity is documentation. If the software code itself is considered as a specialized form of electronic document, all visible outputs are in fact documentation.

It follows that this Standard focuses on requirements for the structure and content of the documentation produced.

Software is used for the implementation of highly complex functions. The ability to deal with a high level of complexity in a flexible way makes software an essential and increasing part of space segment and ground segment products. In space systems, software engineering is found at all levels, ranging from system level functions down to the firmware of a space system.

Therefore the requirements engineering process, in which the software requirements and specifications are defined, has a special emphasis in this Standard. The software requirements and architecture engineering process consumes a large and often underestimated amount of effort in the development of software for space systems.

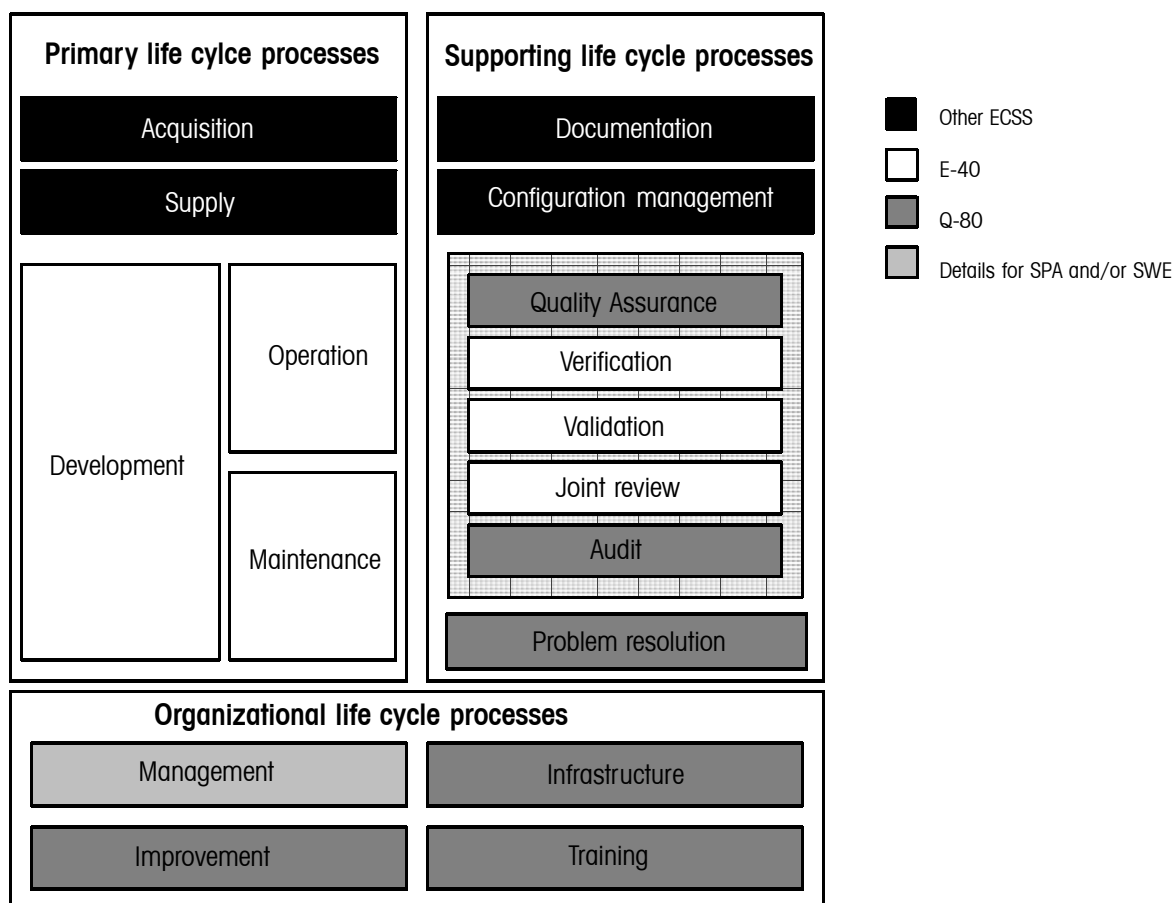
As a result of the complexity of the functional and performance requirements, it also follows that special measures and emphasis apply for software verification and validation, especially for flight software. The functions assigned to software can be critical to the space mission.

The maintenance of software for space systems creates additional constraints and conditions because the operational lifetime can exceed those of general computer software products. For the space segment, this is further complicated by the fact that software in general is the only part of the space segment that undergoes

major maintenance and repair, sometimes even redesign, after launch. In extreme cases, the space system mission itself is redesigned, implementing new flight software after launch. Ground segment software is similarly characterized.

This Standard is complemented by ECSS-Q-80 Space product assurance — Software product assurance, with product assurance aspects. Together the two standards either define or refer to the definition of all software relevant processes for space projects.

The coverage of all software life cycle processes by the different ECSS Standards is illustrated in Figure 1.



**Figure 1: Software life cycle processes in ECSS Standards**

## 4.2 Overview of space system software engineering processes

### 4.2.1 General

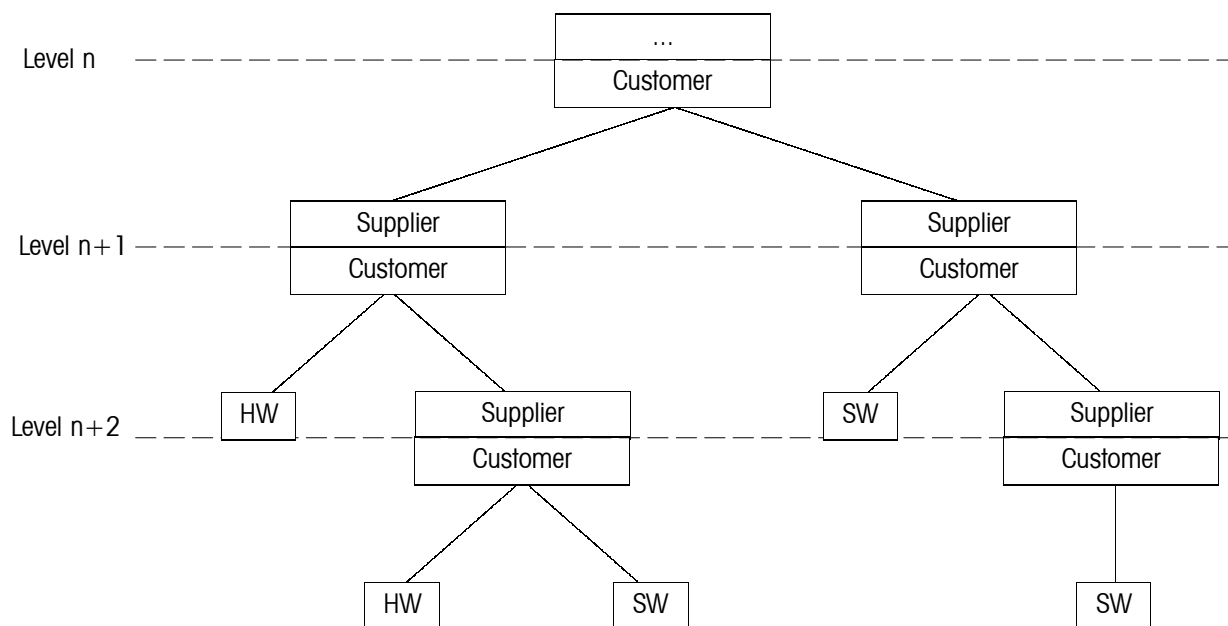
The software engineering processes regulated by this Standard are based on the definitions and requirements given in the ECSS-M series (in particular M-20, M-30, M-40 and M-50), and the general engineering process requirements of ECSS-E-00. These requirements have been used to define the top level software engineering processes. This general framework defines the processes (that are later treated in detail in the following subclauses) and the top level interface between the software engineering processes and other space development processes.

**NOTE** The software analysis process (as a general engineering process defined in the ECSS-E standards) is invoked by the requirements and design engineering processes. No separate output is produced by this process. The results produced

by the analysis process are integrated with the requirements and design engineering outputs.

The fundamental principle of this Standard is the ‘customer-supplier’ relationship, assumed for all software developments. The organizational aspects of this are defined in ECSS-M-20. The customer is, in the general case, the procurer of two strongly associated products: the hardware and the software for a system, subsystem, set, equipment or assembly (see ECSS-E-00). The concept of the “customer-supplier” relationship is applied recursively, i.e. the customer may be a supplier to a higher level in the space system, as shown in Figure 2. The software customer therefore has two important interfaces. He first interfaces with its software and hardware suppliers in order to adequately allocate to them, through the functional analysis, functional and performance requirements. He then assumes a supplier role to interface with its customer, ensuring that higher level system requirements are adequately taken into account.

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

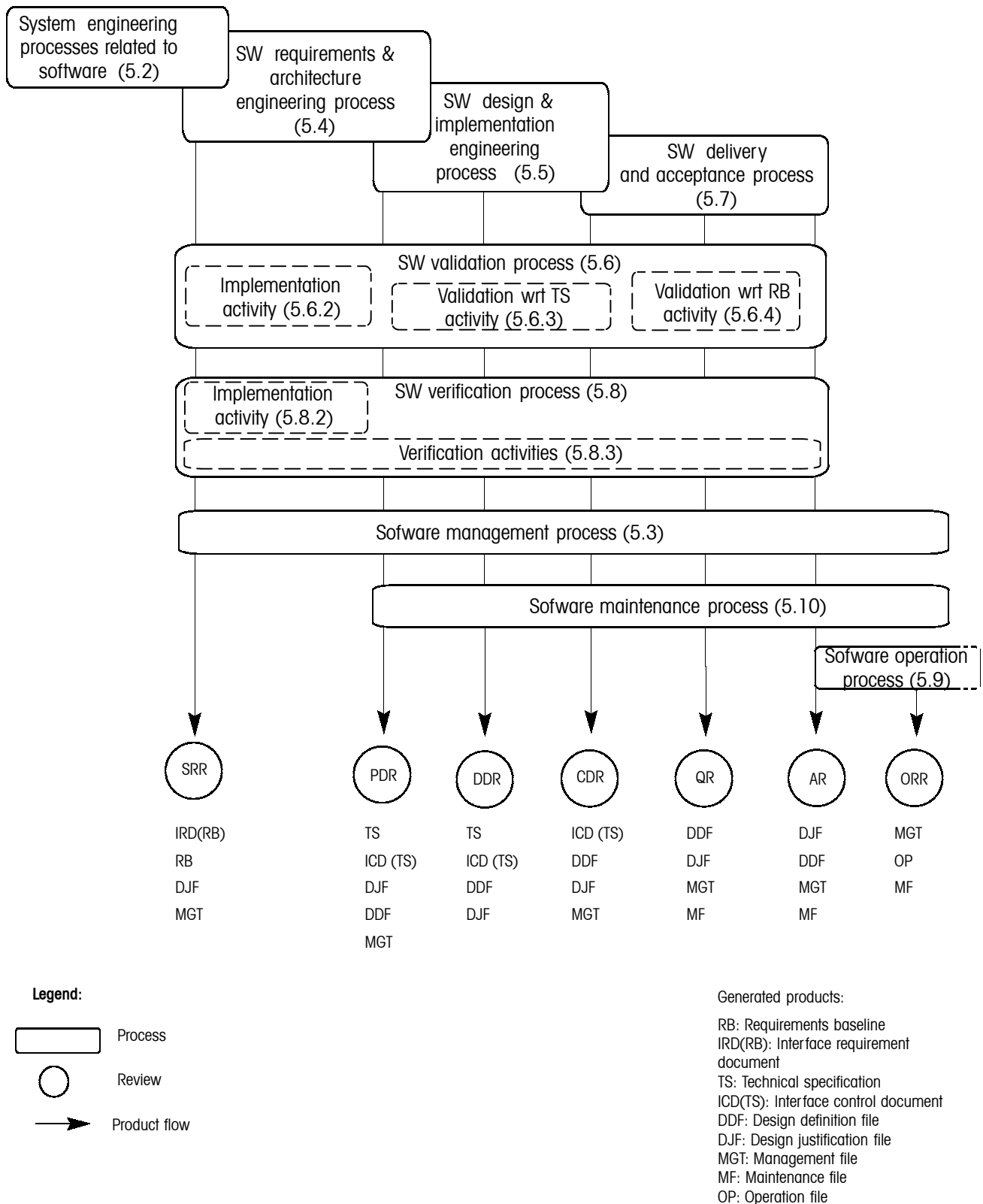


**Figure 2: The recursive customer - supplier model**

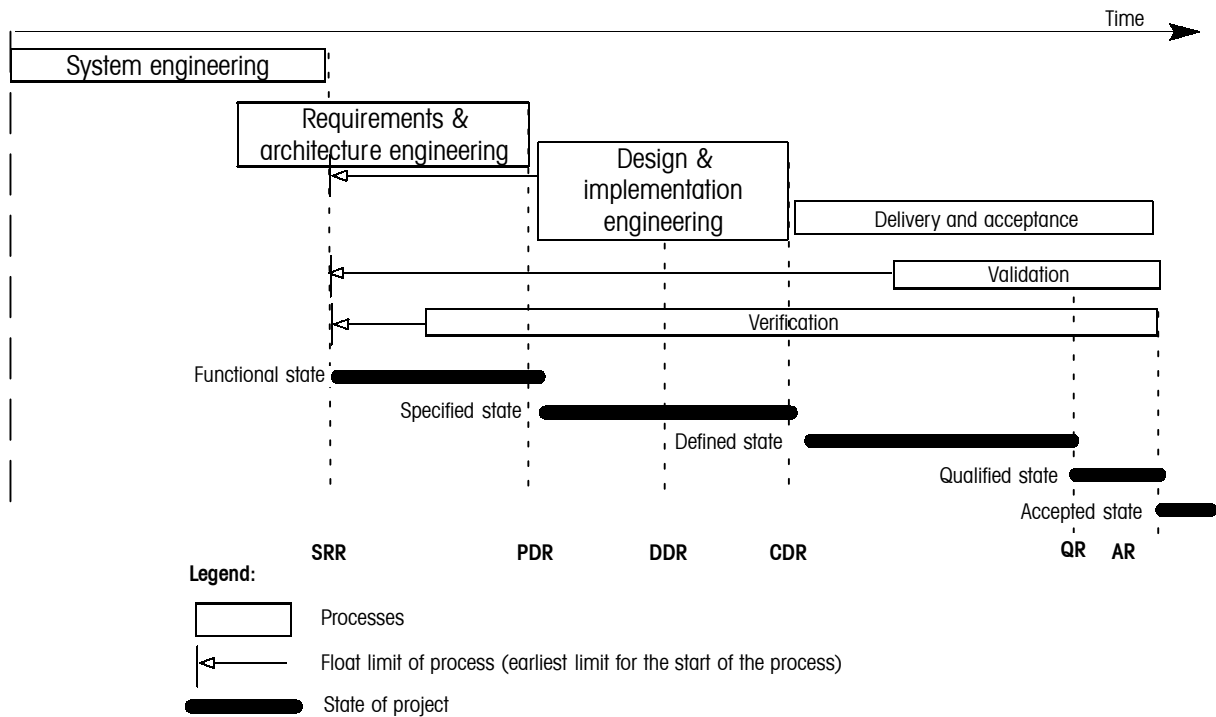
Reviews are the main interaction points between the customer and supplier. The reviews relevant to the software engineering processes are the SRR, PDR, DDR, CDR, QR and AR, as defined by ECSS-M-30. All reviews are applicable to software. The reviews occur at different levels in the customer-supplier hierarchy and are sequenced according to the overall system level planning. This Standard is designed to be applied at any level, without explicit assumptions of how these reviews are integrated with other reviews in the development of a space system. The term system in this Standard is meant as system or subsystem at any decomposition level. An overview is shown in Figure 3. The commonly designated mission phases (e.g. 0, A, B) are used for the overall mission phases, and play no direct role in the software engineering activities as such. This means that the software engineering processes, together with their reviews and milestones as

defined in this Standard, are planned in relation to the immediate higher level development processes.

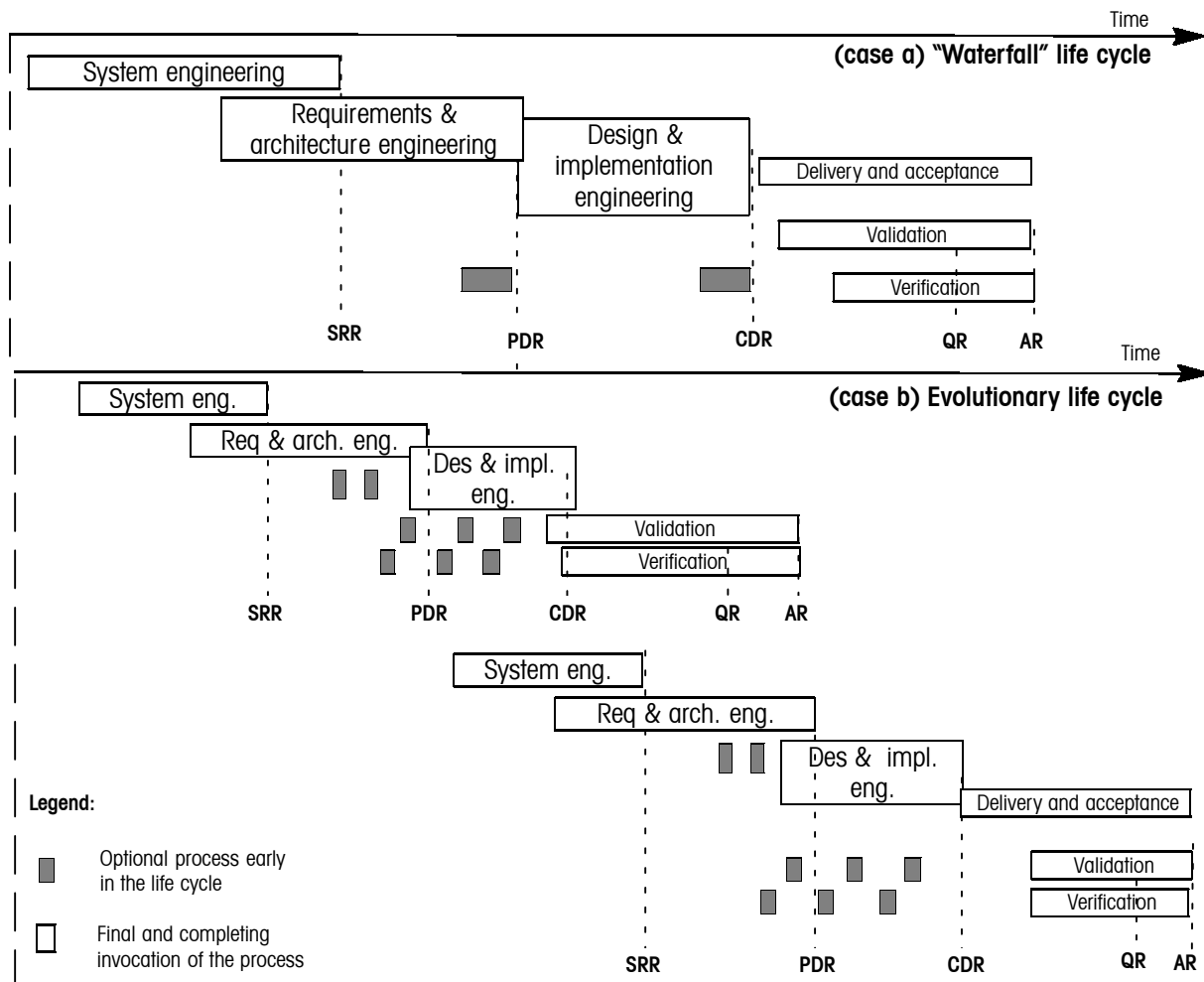
The notion of engineering processes is fundamental to this Standard, as the processes provide the means to describe the overall constraints and interfaces to the software engineering processes at system level, and at the same time provide the necessary freedom to the supplier to implement the individual activities and tasks implied by the processes. The freedom given to the supplier to implement the engineering processes is especially important for software engineering, because of the requirement to organize the work in accordance with a well defined software life cycle. There is a requirement to accommodate different types of software life cycles, both for reasons of efficient organisation of the work and also for reasons related to competitiveness and choice of software engineering technology. Different software life cycle types can be accommodated within the requirements in this Standard. Figure 4 illustrates the constraints. Figure 5 shows examples of variations within these constraints.



**Figure 3: Overview of the software life cycle processes**



**Figure 4: Process constraints**



**Figure 5: Accommodation of different software life cycles**

#### **4.2.2 Software management process**

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

The M-branch of the ECSS standards defined how to manage the software project.. This process complements the M standards for software-specific issues.

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

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

#### **4.2.3 System engineering processes related to software**

The system engineering processes produce the information for input to system requirements review (SRR). This establishes the functional and the performance requirements baseline (RB) of the software development.

#### **4.2.4 Software requirements and architecture engineering process**

The software requirements and architecture engineering process consists of the elaboration of the technical specification (TS), which is the supplier's response to the requirements baseline. This process can start in parallel or after the elaboration of the requirements baseline. The software product tree is defined by this process. The technical specification contains a precise and coherent definition of functions and performances for all levels of the software to be developed. The preliminary interface control document (ICD) is generated by this process.

During the software requirements and architecture engineering process, the result of all significant trade-offs, feasibility analyses, make-or-buy decisions and supporting technical assessments are documented in a design justification file (DJF).

The software requirements and architecture engineering process is completed by the preliminary design review (PDR). The input to the PDR is the technical specification, preliminary ICD and the DJF. The software architectural design is reviewed at the PDR. Additional reviews of the software architecture at the customer's initiative can be specifically included in the organization of the project.

The state of the software development after PDR is called "specified state".

#### **4.2.5 Software design and implementation engineering process**

The "design and configuration engineering process" mentioned in ECSS-E-10 is referred to, in software developments, as the "design and implementation engineering process".

This process does not start before the SRR. It can start before the PDR, but it is after the PDR, when the results of the requirements and architecture engineering process are reviewed and baselined, that are used as inputs to the design and implementation engineering process.

The main output of this process is the design of the software items identified in the software product tree. It is provided in response to the technical specification and the design justification file. All elements of the software design are documented in the design definition file (DDF). The DDF contains all the levels of design engineering results, including software code listings.

The rationale for important design choices, and analysis and test data that show that the design meets all requirements, is added to the DJF by this process. The results of this process are the input to the critical design review (CDR). The CDR signals the end of the design activities. For large software projects, all software subsystems undergo a CDR before they are integrated with the next highest level in the system hierarchy. Large software developments are partitioned into smaller manageable projects that are managed like any other subsystem development in space projects.

Finally this process produces also the coding, unit testing and integration testing of the software product. All elements of the testing activities are documented in the design justification file (DJF).

The state of the software project after CDR is called “defined state”.

#### **4.2.6 Software validation process**

The software validation process can start any time after the SRR.

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

The result of this process is included in the DJF.

This process is implemented before the PDR to basically produce a software validation plan. This includes the validation activity with respect to the technical specification (to be held before the CDR) and the validation activity with respect to the requirement baseline (to be held before the QR and possibly repeated at AR).

This process includes a qualification review (QR) with the DJF as an input. This process can also include a test readiness review (TRR) to verify that all test facilities and test cases are available before each test campaign.

The state of the software project after QR is called “qualified state”.

#### **4.2.7 Software delivery and acceptance process**

The software delivery and acceptance process can start after the CDR and when the software validation activity with respect to the technical specification is complete.

This process includes an acceptance review (AR), with the DJF as input. The acceptance review is a formal event in which the software product is evaluated in its operational environment. It is carried out after the software product is transferred to the customer and installed on an operational basis. Software validation activities terminate with the acceptance review. The state of the software project after AR is called the “accepted state”.

#### **4.2.8 Software verification process**

The software verification process can start any time after the SRR.

This process is intended 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 inputs.

The result of this process is included in the DJF.

This process is implemented before the PDR to basically produce a software verification plan. This process includes all the verification activities of all the expected outputs of the development activities

NOTE The term “qualification engineering” is often used synonymously with the term “verification engineering” in projects delivering hardware. For the sake of clarity, “qualification engineering” is used in this Standard to denote “the total set of verification and validation activities”. This is consistent with other ECSS Standards outside the software engineer-



ing discipline, and to avoid confusion with the general verification engineering activities that are invoked in many points in software projects.

#### **4.2.9 Software operation process**

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

General requirements for operations are found in ECSS-E-70 .

#### **4.2.10 Software maintenance process**

This separate process is started after the completion of the AR but the process implementation is started before the QR.

This process is activated when the software product undergoes any modification to code or associated documentation as a result of correcting an error, a problem or implementing an improvement or adaptation. The process ends with the retirement of the software product.

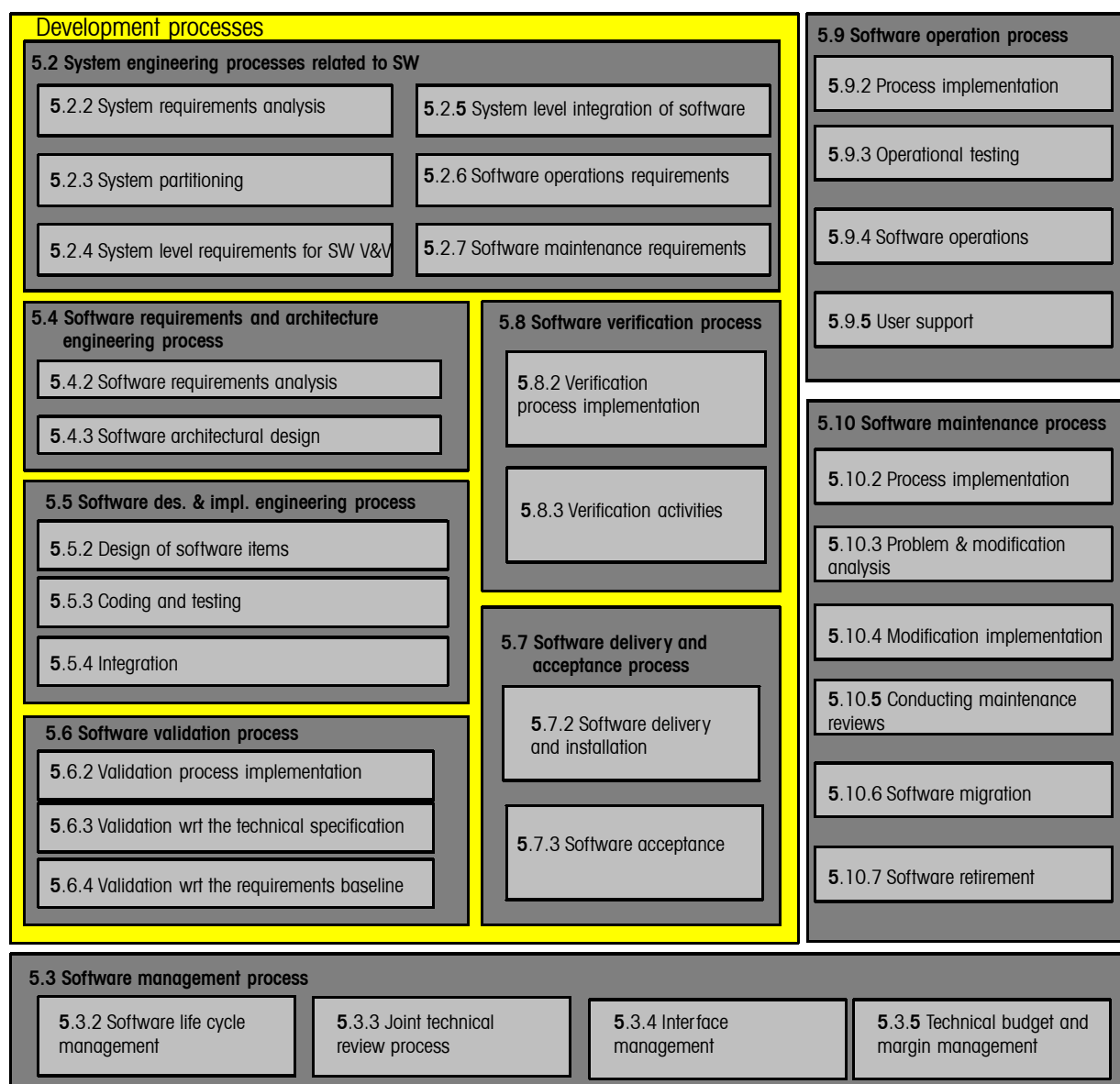
### **4.3 Organization of this Standard**

This Standard contains one normative clause (clause 5) which is organized with respect to the software processes and activities breakdown described in Figure 7. Each process includes activities which are themselves decomposed into a list of one single or several requirements (subclauses). Those requirements are either core requirements applicable for any space system software engineering activity, or special requirements applicable only when the software project or product to be developed addresses disciplines or application areas like flight software, software reuse, man-machine interfaces or real-time software. Descoping of requirements can be done by the tailoring process (see Annex C) taking into account project or software product characteristics.

Software documentation summaries are included in annex A.

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 engineering process description.

The organization of the general requirements of this Standard is reflected in detail in Figure 6.



**Figure 6: Structure of this Standard**

## 4.4 Relation to other ECSS Standards

### 4.4.1 General

This subclause 4.4 describes how this Standard interfaces with other ECSS series, namely the ECSS-Q series of standards (product assurance) and the ECSS-M series of standards (project management).

### 4.4.2 Software product assurance

Requirements on software product assurance are defined in ECSS-Q-80, which is the entry level document of the ECSS-Q series (product assurance) for software projects.

The ECSS-Q-80 Standard covers all aspects of space software product assurance including the implementation aspects of the software product assurance process, and both software process and product-related assurance activities.

It defines the scope of the space software product assurance process and its interfaces with management, engineering and other system-level product assurance activities, which are addressed in the management (-M), engineering

(-E) and product assurance (-Q) branches of the ECSS System, and explains how they apply in the software product assurance process.

### **4.4.3 Software project management**

#### **4.4.3.1 Introduction**

ECSS-M Standards define the requirements applicable to the management of space projects. Subclauses 4.4.3.2 to 4.4.3.9 describe how the ECSS-M Standards apply to the management of software projects.

In addition, requirements that cannot be found in the M-series, because they are specific to software project management, are provided in subclause 5.3.

#### **4.4.3.2 ECSS-M-00: Policy and principles and ECSS-M-00-03: Risk management**

ECSS-M-00 is a top-level document which defines project management principles and general requirements applicable to all aspects of a space project including software.

Risk management is covered by ECSS-M-00-03. Some risk factors, such as exceeding the assigned memory budget or CPU load, are specific to software.

The terms “customer” and “supplier” used in this Standard are defined in ECSS-M-00A subclause 5.2.

#### **4.4.3.3 ECSS-M-10: Project breakdown structures**

The provisions of ECSS-M-10 apply to software, taking account of the specific features of the software.

The products of a software project are usually documents (including code), but can also include computer devices in the case of software intensive systems.

“Model matrix” in ECSS-M-10A, subclause 5.2, is concerned with material models and therefore is not relevant to software.

#### **4.4.3.4 ECSS-M-20: Project organization**

ECSS-M-20 provides a clear definition of the role and responsibility of each party in the project. ECSS-M-20 covers the requirements for software projects.

#### **4.4.3.5 ECSS-M-30: Project phasing and planning**

ECSS-M-30 defines the phasing and planning requirements for an entire space project, but some requirements also affect software development, because they are specified in ECSS-M-30 as applicable at any level of the project organization.

Project phases as defined in ECSS-M-30 are top-level (mission) phases, used to structure the whole space project. They do not apply recursively to software development. They are not the phases that are defined to give structure to software development life cycles, and for which no specific definition is requested in this Standard.

Similarly, the reviews as defined in ECSS-M-30 do not apply directly to software, even though the concept of review applies recursively to all levels of a space project.

The terms “SRR”, “PDR”, “CDR”, “QR” and “AR” are defined in ECSS-M-30, and these are reused to define joint technical reviews for a software development as described in subclause 4.2 of this Standard.

These reviews are synchronized with higher level reviews in a way which is project dependent. Interface requirements are also given for particular types of software. Requirements concerning phasing and reviews, and which are specific to software are given in subclause 5.3.

#### **4.4.3.6 ECSS-M-40: Configuration management**

The requirements, also for software developments, are contained in ECSS-M-40. One facet of software configuration management is that all configuration items can be regarded as documents (even code). Therefore, the software configuration management can easily be automated.

#### **4.4.3.7 ECSS-M-50: Information/documentation management**

Information and documentation management are particularly performed to ensure the accessibility of information to all parties of the project and to ensure the coherence of this information. This also applies to software projects. The relevant requirements are those in ECSS-M-50.

#### **4.4.3.8 ECSS-M-60: Cost and schedule management**

ECSS-M-60 contains requirements on software projects, although requirements on schedule management are more directly applicable to software than costing requirements.

#### **4.4.3.9 ECSS-M-70: Integrated logistic support**

ECSS-M-70 is mainly of concern to large or software-intensive systems.

### **4.4.4 Engineering**

#### **4.4.4.1 ECSS-E-00: Policy and principles**

ECSS-E-00 contains the basic rules and overall principles to be applied to all engineering activities during the performance of a space project. It addresses the establishment, based on customer needs, of mission objectives, requirements, and specifications for space systems, and the design, definition, production, verification, operation, and eventual disposal of the systems themselves. It defines the scope and interfaces of these activities relative to the domains of management and product assurance that are addressed in the management (- M) and product assurance (- Q) branches of the ECSS system, and explains how they can be applied in different ways depending on the type of space system concerned.

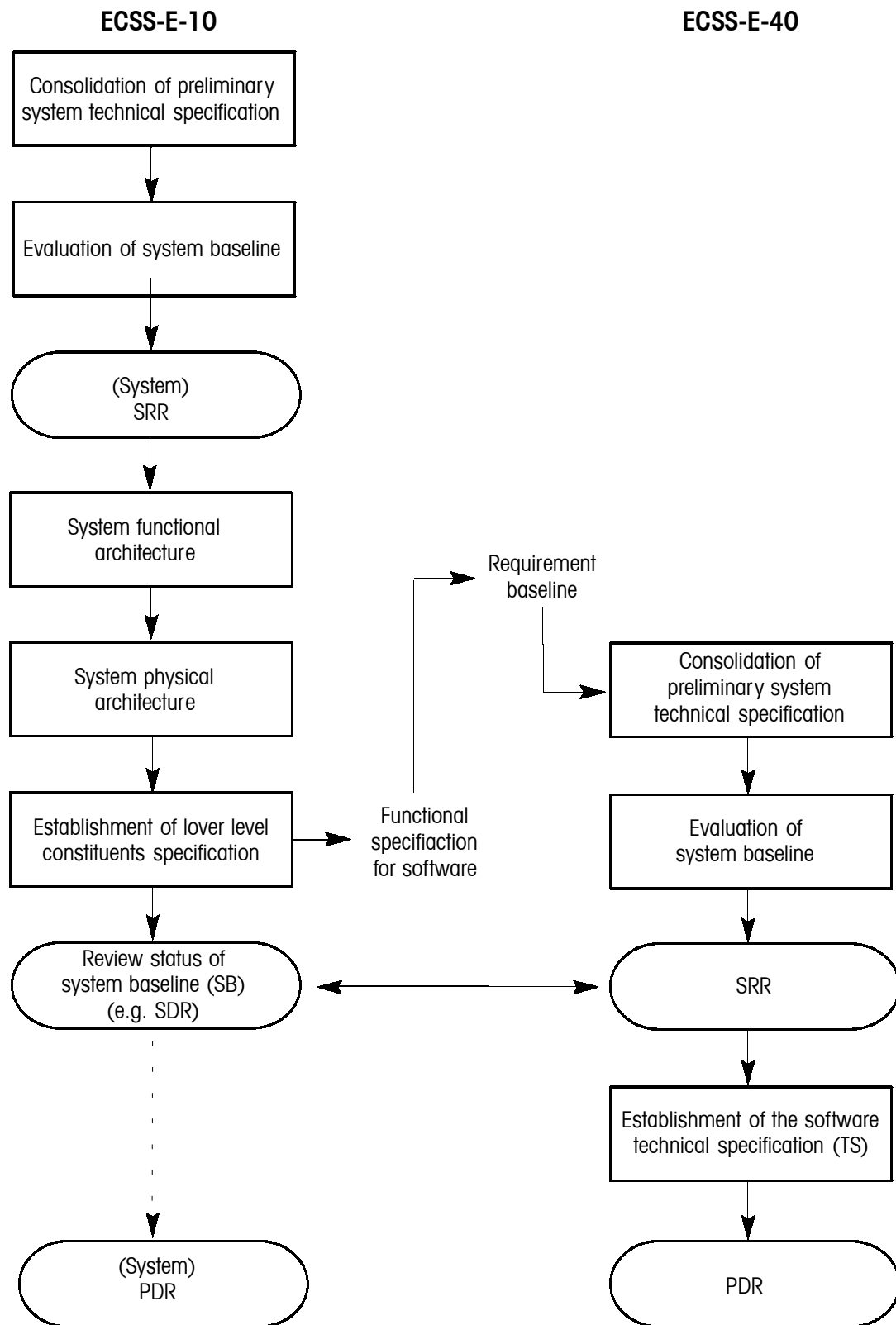
#### **4.4.4.2 ECSS-E-10: System engineering**

ECSS-E-10 is intended to guide the development of systems (including hardware, software, man-in-the-loop, facilities and services) for space applications. It specifies implementation requirements for the responsible system engineering organization consistent with the assumption that the system engineering process defined in standard ECSS-E-10 is applied.

The link between ECSS-E-40 and ECSS-E-10 is reflected in the clause 5.2 of this standard. It is called "System engineering processes related to software". This clause 5.2 complements ECSS-E-10 for the specific software activities to be performed at system level by the customer. Note that these activities can be delegated by the customer to the supplier. These specific activities are performed during system phase B. The ECSS-E-10 Phase B flow chart indicated in particular the following activities:

- Consolidation of the system operation concept and related functional architecture,
- Definition and justification of the system baseline physical architecture, and
- Establishment of lower level constituents specifications

The latter provides the (system level) functional specification for the software that is called in this ECSS-E-40 standard the requirement baseline (as defined in ECSS-E-10)



**Figure 7: ECSS-E-10 and ECSS-E-40 relationship**

In the ECSS-E-10 activity (Review status of system baseline and Associated Plans), the system engineer takes into account the subsystem views to consolidate the system baseline and can formalise it into the system SDR. This review is the equivalent of the software SRR in ECSS-E-40.

The ECSS-E-40 activity “evaluation of system baseline” verifies that the specific software activities at system level described in clause 5.2 are actually taken into account. This is formalized at the software SRR as mentioned in subclause 5.2.4.5, the software being viewed now as a (lower-level) system.

NOTE The practical implementation of this scheme is a matter of ECSS-E-40 standard tailoring (e.g. the delegation of the customer software activities to the supplier)

#### **4.4.4.3 ECSS-E-70: Ground systems and operations**

ECSS-E-70 provides a high level description of all ground segment elements, the domain specific aspects of the associated engineering processes and defines related guidelines and requirements.

## **4.5 Tailoring of this Standard**

The general requirements for selection and tailoring of applicable standards are defined in ECSS-M-00-02.

There are several drivers for tailoring, such as dependability and safety aspects, product quality objectives, software development constraints and commercial considerations.

Tailoring for software development constraints takes into account technical, operational and management factors. See annex C for details.

---

## 5

# Requirements

## 5.1 Introduction

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

Each requirement can be identified by a hierarchical number. The text of the requirement is followed, where necessary, by further explanation of the aim. For each requirement, the associated output is given in the output section. With each output (e.g. “a.”, “b.”), the destination (document) of the output is indicated in brackets together with the corresponding review. For example: “[DDF, DJF; QR]” denotes an output to the design definition file and the design justification file. The output in this example is requested for the qualification review.

Some requirements depend on the nature of the software. This is explicitly mentioned in the requirements. They include:

- Flight software

The flight software calls for special engineering requirements, due to the highly specialized environment and because the software implements functions that directly relate to space system dependability. Flight software can be reactive or interactive.

Detailed software engineering guidelines for flight software are found in ECSS-E-40-01

- Software reuse

Some specific subclauses are applicable in the software engineering processes for projects where:

- it is intended to reuse the software products being developed for other space projects;
- it is intended to reuse software products from other space projects and third-party “commercial off-the-shelf” products are intended to be part of the software product.

- Man-machine interface

Software projects that include the development of a significant interactive direct interface to a human user or operator lead to the involvement of the specialized software engineering and human factor disciplines covering this field.

The reason for the special subclauses is that it is not feasible to specify or design modern MMI technology (e.g. graphical user interfaces, and multi-layered choice menus) using conventional software engineering documentation methods. The non-linear and multi-dimensional nature of modern MMI cannot be described adequately using only two-dimensional documents that by nature are linear in structure. This is very similar to other systems with significant human factor requirements, such as cars, airplanes, and buildings. In those cases a mock-up or model is implemented during the requirements and architecture engineering process. An analogous approach in software engineering applies for software with extensive human interaction requirements.

For any MMI development a scenario-driven requirements analysis can be performed. The term MMI used in the corresponding subclauses also includes customization of COTS supplied MMI.

- Real-time software

Real-time software have strong time constraints to respect (e.g. handling of input data before its loss of validity) that justify the existence of some special subclauses (e.g. software dynamic design, behaviour modelling and schedulability analysis)

## 5.2 System engineering processes related to software

### 5.2.1 Introduction

This subclause 5.2 describes activities that are under the customer's responsibility. The customer is responsible for the delivery of a system in which the developed software is integrated (refer to the recursive customer-supplier model described in 4.2).

The customer activities described here are only those that introduce additional requirements particular for software development:

- system requirement analysis,
- system partitioning,
- system level requirements for software verification and validation,
- system level integration of software,
- software operations requirements, and
- software maintenance requirements.

System level documentation is a prerequisite for the requirements engineering of the software. The requirements given in this subclause ensure the completeness and correctness of the customer's system level documentation and establish a complete and verified requirements baseline for the software project.

### 5.2.2 System requirements analysis

#### 5.2.2.1 System requirements specification

System requirements shall be derived from an analysis of the specific intended use of the system to be developed and documented.

EXPECTED OUTPUT: *a. Functions and performance requirements of the system [RB; SRR];*  
*b. Interface requirements [IRD(RB); SRR];*  
*c. Design constraints [RB; SRR];*  
*d. Identification of lower level software engineering standards [RB; SRR] (see ECSS-Q-80B subclauses 6.3.2 and 6.3.3);*



- e. *Verification and validation product requirements [RB; SRR].*

#### **5.2.2.2 System and functional criticality analysis**

System and software criticality analysis shall be performed in accordance with ECSS-Q-80B subclauses 6.2.2 and 6.2.3

EXPECTED OUTPUT: *Overall safety and reliability requirements of the software to be produced [RB; SRR].*

#### **5.2.2.3 Establishment of the need for a MMI mock-up**

For software with interface to human operators, the customer shall, based on the complexity and requirements of the MMI, determine if a software mock-up of the MMI is requested to support the requirements and architecture engineering process.

EXPECTED OUTPUT: *MMI software mock-up requirements [RB; SRR].*

#### **5.2.2.4 MMI standards definition**

The customer shall determine if general MMI standards shall be applicable to the software project and include these requirements in the requirements baseline.

AIM: To ensure that appropriate guidelines, and style-guides are selected for projects in cases where, for example, a common MMI style and functionality applies for several suppliers' products.

EXPECTED OUTPUT: *MMI general requirements [RB; SRR].*

### **5.2.3 System partitioning**

#### **5.2.3.1 Introduction**

- a. As a part of the system design process, a physical architecture and design (including HW, SW and human operations) of the system shall be derived.

NOTE 1 This is called top-level partitioning of the system.

NOTE 2 This system design is derived from an analysis of the requirements on the system and its functions.

EXPECTED OUTPUT: *System design [DDF-system level; SRR].*

- b. Conformity of the system design with all system requirements shall be verified.

EXPECTED OUTPUT: *System design to system requirements conformance [DJF-system level; SRR].*

- c. All system requirements shall be allocated and traceable to the different system design partitions.

EXPECTED OUTPUT: *System requirements to system design traceability [DJF-system level; SRR].*

#### **5.2.3.2 System partitioning**

- a. A top-level partitioning of the system shall be established.

EXPECTED OUTPUT: *Software-hardware interface requirements [IRD(RB); SRR].*

- b. The system partitioning shall identify items of hardware, software and human operations, ensuring that all the system requirements are allocated to items.

EXPECTED OUTPUT: *Traceability to system partitioning [DJF; SRR].*

- c. Hardware configuration items, software configuration items, and human operations shall be subsequently identified from these items.

EXPECTED OUTPUT: *System partition with definition of items [RB; SRR].*

- d. The system partitioning and the system requirements allocated to the individual items shall be documented.

EXPECTED OUTPUT: *System configuration items list [RB; SRR].*

## **5.2.4 System level requirements for software verification and validation**

### **5.2.4.1 Introduction**

The general ECSS approach to the verification process is described in ECSS-E-10A clauses 4 and 5, covering both verification and validation activities.

### **5.2.4.2 Qualification engineering requirements**

The customer shall adapt the requirements for qualification engineering given in subclauses 5.6, 5.7 and 5.8 to system level requirements.

AIM: To identify the customer's verification and validation process requirements at system level, and to prepare for software acceptance and software integration by introducing the corresponding verification and validation process requirements in the requirements baseline.

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

### **5.2.4.3 Software validation requirements at system level**

The customer shall include requirements for validation of all elements of the software at system level, including validation at mission level.

AIM: To ensure that the software is validated at system level with realistic mission data and operational environments, and to minimize the functions that can only be validated by actual flight. This is because in general no prototype flights are possible, and then mission success imposes the software to be operational at the first flight.

EXPECTED OUTPUT: *a. Functional requirements for support to system and mission level validation [RB; SRR];*

*b. Installation and acceptance requirements for the delivered software product at the operational and maintenance sites [RB; SRR].*

### **5.2.4.4 Requirements baseline verification**

The customer shall verify the requirements baseline, considering the following:

- In cases where the customer's product is an integrated hardware and software product, verification shall be performed in conformance with the ECSS system engineering standards.
- In cases where the customer's product is a software product, the customer shall apply this Standard in his role as "supplier" at a higher level in the product tree.

EXPECTED OUTPUT: *Requirements justifications [DJF-system level; SRR].*

NOTE The output is not part of the customer-supplier interface for the software engineering processes, and is therefore not part of any milestone inputs. Instead the output is part of the customer's own system DJF, and is used only by the customer in his role as supplier at the next higher level in the product tree. The output is mentioned here for completeness only.

#### 5.2.4.5 System requirements review

The customer shall conduct a system requirements review (SRR) in accordance with subclause 5.3.2.6.

EXPECTED OUTPUT: *SRR milestone report [RB; SRR].*

### 5.2.5 System level integration of software

#### 5.2.5.1 Identification of observability requirements

- a. If a software product is integrated into a system, all software observability requirements to facilitate the software integration shall be specified by the customer.

EXPECTED OUTPUT: *Software observability requirements [RB; SRR].*

- b. Observability data shall include the data for the system observability and shall take into account the constraints imposed by the computer such as the bandwidth allocation and the overloading of the processor.

NOTE 1 Observability requirements can impair the performance requirements (e.g. computer throughput, bandwidth, and ground exploitation rate of telecommands). This has an impact when specifying the observability requirements (e.g. considering the need of oversampling).

EXPECTED OUTPUT: *System observability requirements [RB; SRR].*

- c. The customer shall specify the system observability data individually to be completed by the supplier later in the technical specification.

EXPECTED OUTPUT: *System observability requirements [RB; SRR].*

- d. When specifying the observability requirements, the customer shall trade off the software visibility with the risk of activating undesirable code (e.g. on-board).

NOTE 2 For flight software, the software observability requirements, considered only during the integration, and not during flight, can result into some deactivated code.

EXPECTED OUTPUT: *System observability requirements [RB; SRR].*

#### 5.2.5.2 Control and data interfaces for system level integration

If the software is integrated into a system, all the interfaces between the software and the system shall be specified by the customer, including the static and dynamic aspects, for nominal and degraded modes (e.g. behaviour in case of failure).

NOTE 1 The external interfaces, specific to software integrated in a system, can be:

- software interfaces with other software in the system (operating system, files, database management system or other applications software);
- hardware interfaces to the specific hardware configuration; and
- communication interfaces (particular network protocol for example).

NOTE 2 Flight software is in general integrated with highly specialised processors and electrical equipment. The IRD and ICD therefore have a special importance and are controlled separately to ensure consistent design throughout the hardware and software life cycle.

EXPECTED OUTPUT: *System level interface requirements [IRD(RB); SRR].*

#### 5.2.5.3 Data medium requirements for integration

The customer shall identify the interface data medium and prepare the requirements accordingly.

EXAMPLE The interface data can be defined and structured in such a way that it can be automatically acquired by the SDE supplier. Trade-offs can be performed, taking into account the number of software packages in the system, the evolution of interface data, and the number of interface data sets.

EXPECTED OUTPUT: *System level data interfaces [IRD(RB); SRR].*

#### 5.2.5.4 System database

The customer shall specify a system database to ensure the consistency of common data, considering the following additional aspects:

- Specification of the system database use for the supplier. For instance, the database can be used to produce automatically configured software (e.g. generation of tables, constant data, initial values).
- Specification of the system database size, the possible modification that can be foreseen after the acceptance review, and the accepted impact in terms of software maintenance.

NOTE System database can be, according to ECSS-E-10 and ECSS-E-70, e.g. a mission-operations database, an operational database of ground elements, a space segment database or a spacecraft database.

EXPECTED OUTPUT: *System database specification (content and use) [RB; SRR].*

#### 5.2.5.5 Identification of development constraints

The customer shall define specific development constraints on the supplier to support the integration of the software into the system.

NOTE When the software is integrated into a system, the customer can check for applicability of some harmonization constraints such as: specification of the operating system to be used, specification of COTS to be used (e.g. database and MMI generator), and specification of the SDE to be used.

EXPECTED OUTPUT: *Development constraints [RB; SRR].*

#### 5.2.5.6 Definition of constraints for software to be reused

The customer shall specify the reusability requirements that apply for the development, to enable future reuse of the software.

NOTE When reuse of developed software components is a requirement, the customer specifies the generic application domain of these components. This can, for example, include requirements on software architecture for given target computers and operating systems, the interfaces required for reuse and the level where reuse is required (e.g. function, sub-system, and code modules).

EXPECTED OUTPUT: *Requirements for 'design for reuse' [RB; SRR].*

#### 5.2.5.7 Identification of customer's input for software integration into the system

The customer shall identify and plan the specific inputs to be provided to the supplier to support the integration of the software into the system, in accordance with the overall projects constraints with appropriate documentation in the requirements baseline.

NOTE When the software is integrated into a system, the customer can provide the supplier with specific inputs for validating the software in a representative environment. These inputs can be: a breadboard or computer model, and a simulator of the hardware and software environment.

EXPECTED OUTPUT: *System level integration support products [IRD(RB); SRR].*

#### **5.2.5.8 Identification of supplier's outputs for software integration into the system**

The customer shall identify and plan the specific outputs to be delivered by the supplier to support the integration of the software into the system, and shall prepare the requirements baseline accordingly.

NOTE When software is integrated into a system, some prototype versions or intermediate versions can be requested by the customer to prepare the integration. The functionalities and delivery dates for each of these versions are defined by the customer.

EXPECTED OUTPUT: *System level integration preparation requirements [IRD(RB); SRR].*

#### **5.2.5.9 Planning of supplier support to system integration**

The customer shall plan the support from the software supplier in order to integrate the software at system level.

NOTE This can include activities such as: training, maintenance, configuration and test support. This is reflected in the supplier's plans in the management file.

EXPECTED OUTPUT: *System level integration support requirements [MGT; SRR].*

### **5.2.6 Software operations requirements**

#### **5.2.6.1 Phasing and management**

Since software products are an integrated part of the space system, the phasing and management of operations shall be determined by the overall system requirements and shall be applied to the operations of software products.

EXPECTED OUTPUT: *Operational plan [OP; ORR].*

#### **5.2.6.2 System requirements definition for software operations**

The customer shall establish system requirements for the operation of software products.

NOTE The supplier's response is provided in releasing the operational plan for execution as established in subclause 5.9.

EXPECTED OUTPUT: *Software operations requirements [RB; SRR].*

### **5.2.7 Software maintenance requirements**

#### **5.2.7.1 System requirements related to maintenance**

The customer shall establish system requirements for the maintenance of software products.

NOTE The supplier's response is agreed with the customer in the system requirements review (SRR), intended to release the maintenance plan for execution as established in subclause 5.10.

EXPECTED OUTPUT: *Software maintenance requirements [RB; SRR].*

#### 5.2.7.2 Definition of in flight modification capability for flight software

The customer shall specify the requirements to perform software modifications in flight, when this capability is identified for flight software.

NOTE Due to the long lifetimes often encountered with flight software, special requirements also exist to ensure that the supporting tools (e.g. compilers, engineering tools and in-flight modification tools) can support the in-orbit reprogramming during the specified lifetime.

EXPECTED OUTPUT: *Requirements for in flight modification capabilities [RB; SRR].*

### 5.3 Software management process

#### 5.3.1 Introduction

Most of the specific requirements for the management and control of space systems software projects exist in the ECSS-M series of documents. They are not repeated here. In addition, the software product assurance requirements described in ECSS-Q-80 are also used for the control of space systems software projects. Management plans are produced in relation with the following activities:

- development;
- configuration and documentation management;
- verification and validation;
- maintenance;
- quality assurance on process and product.

The requirements described in this subclause 5.3 define the engineering and control of software development in a space systems project, and they bridge the gap between the other ECSS Standards mentioned above and the software engineering activities in space projects.

The management and control tasks described in this subclause are:

- software life cycle management,
- joint technical review,
- interface management, and
- technical budget and margin management.

The requirements in this subclause 5.3 apply to any type of software in a space project.

As defined in more detail in the following subclauses, the software undergoes the overall software milestone reviews SRR, PDR, DDR CDR, QR and AR as a minimum. A DDR is also requested for flight software in accordance with subclause 5.3.2.9. The customer can request further reviews (e.g. review of project plans, before the PDR) following requirements mentioned in subclause 5.3.3.

#### 5.3.2 Software life cycle management

##### 5.3.2.1 Definition of software life cycle phases

To assure effective phasing and planning, the software development life cycle shall be broken into phases, each having its associated milestones.

NOTE Detailed guidelines on software life cycle are found in the level 3 Standard ECSS-E-40-04.

EXPECTED OUTPUT: *Definition of the software life cycle phases included in the software development plan [MGT; SRR, PDR].*



**5.3.2.2 Software life cycle identification**

- a. The software supplier shall define and follow a software development and maintenance life cycle in accordance with subclause 4.2, and covering all activities from the statement of requirement to the entry of the software into service.

EXPECTED OUTPUT: *Project software development life cycle definition, included in the software development plan [MGT; SRR, PDR].*

- b. The definition of the life cycle shall be associated with choices of techniques used during the development, operations and maintenance processes (e.g. data base management system, and extensive product reuse), with the risks inherent to the project (e.g. highly changeable specification, and stringent schedule constraints) and with synchronization points with the upper level.

EXPECTED OUTPUT: *Definition of software development, operations and maintenance techniques and identification of project risks, included in the software development plan [MGT; SRR, PDR].*

- c. The choice of software life cycle shall be in accordance with the overall project requirements, and the process model of subclause 4.2 and ECSS-M-30 shall be used.

EXPECTED OUTPUT: *Definition of software life cycle in line with the software and system level processes included in the software development plan [MGT; SRR, PDR].*

**5.3.2.3 Identification of inputs and outputs associated to each phase**

The development life cycle shall define the input and output for each phase and its associated milestones according to 5.3.3.

EXPECTED OUTPUT: *Review plan-milestones (included in the software development plan) [MGT; SRR, PDR].*

**5.3.2.4 Identification of documentation relevant to each milestone**

The output for each phase shall consist of documents in complete or outline versions, including the results of verification of the technical outputs of the phase.

NOTE Milestones are the joint technical reviews at the customer-supplier level (SRR, PDR, DDR, CDR, QR and AR) and internal reviews at the supplier level. The outputs for each milestone are documents submitted for examination and are explicitly listed in the software life cycle definition.

EXPECTED OUTPUT: *Identification of outputs at each milestone (included in the software development plan) [MGT; SRR, PDR].*

**5.3.2.5 Identification of interface between the development and maintenance processes**

The interface between development and maintenance (e.g. documents to be produced, tools to be kept for maintenance) shall be identified for the software life cycle.

AIM: Define and prepare during development input for the maintenance process for the software product. See subclause 5.10

EXPECTED OUTPUT: *Elements of the software maintenance in the software development plan [MGT; PDR].*

**5.3.2.6 Requirements baseline at the SRR**

The customer's release of the requirements baseline shall be included in the material submitted to the SRR.

NOTE The requirements baseline results from a system requirements analysis and a system partitioning conducted by the

customer. It represents the customer's requirements for the software to be developed and it is complemented by the external interfaces requirements of the software.

EXPECTED OUTPUT: *Customer approval of requirements baseline [RB; SRR]*

#### **5.3.2.7 Software technical specification phase**

A software technical specification phase shall be included at the beginning of the development life cycle.

AIM: To establish the technical specification for the project. This is the software supplier's response to the requirements baseline. The technical specification captures all technical requirements for the software product, and it is aimed to establish the technical specification early in the project.

EXPECTED OUTPUT: *a. Software requirements specification [TS; PDR];  
b. Software architectural design [DDF; PDR];  
c. Interface control document [ICD(TS); PDR];  
d. Software architectural design trade-offs [DJF; PDR].*

#### **5.3.2.8 Preliminary design review**

On completion of the technical specification phase, the software supplier shall hold a preliminary design review (PDR) with the customer.

AIM: — Agree with the customer or their representatives that all requirements with respect to the requirements baseline are captured in the technical specification.

— Review the software architecture.

EXPECTED OUTPUT: *Customer approval of technical specification and software architecture [TS, DDF, ICD(TS), DJF; PDR].*

#### **5.3.2.9 Detailed design review**

For flight software, at the end of the detailed design, the software supplier shall hold a detailed design review (DDR) with the customer.

AIM: — Review the detailed design.

— Review the software technical budget status (e.g. CPU and memory).

— Review the completeness and stability of the technical specification requirements. This DDR objective can be implemented in case of evolution of technical specification requirements after the PDR.

NOTE In order to avoid that software items are subject to several reviews, the software elements defined in the general requirements for review at CDR, but actually reviewed at DDR, can be removed from the CDR list.

EXPECTED OUTPUT: *a. Customer approval of the design definition file (software architectural design, detailed design) [DDF; DDR];  
b. Customer approval of the design of software interface and the software integration test plan [DJF; DDR];  
c. Customer approval of the margins and technical budget status [DJF; DDR];  
d. Customer approval of the updated technical specification [TS; DDR].*

#### **5.3.2.10 Critical design review**

a. At the end of the design, the software supplier shall hold a critical design review (CDR) with the customer.



AIM: During the CDR, the design definition file, software user manual and the associated design justification file are reviewed.

EXPECTED OUTPUT: *CDR milestone report [DJF; CDR].*

- b. The completeness of the software validation activities with respect to the technical specification and their relevant products (e.g. test case specification and simulators) shall be reviewed.

EXPECTED OUTPUT: *a. Customer approval of the design definition file (e.g. software architectural design, detailed design, code and software user manual) [DDF; CDR];*  
*b. Customer approval of the design justification file (e.g. software unit test plan, software integration test plan, results of unit and integration tests and results of validation with respect to the technical specifications) [DJF; CDR].*

#### **5.3.2.11 Software verification and validation processes**

Verification and validation shall be completed before AR.

AIM: To ensure, by means of verification and validation processes in a representative environment, that the software product conforms to its technical specification before integration in the system.

EXPECTED OUTPUT: *Software verification and validation activities phasing in the software development plan [MGT; SRR, PDR].*

#### **5.3.2.12 Qualification review**

- a. The software supplier shall hold a qualification review (QR) to verify that the software product meets all of its specified requirements in the requirements baseline.

AIM: To verify that the software meets all of its specified requirements, and in particular that verification and validation process outputs enable transition to “qualified state” for the software products.

EXPECTED OUTPUT: *QR milestone report [DJF; QR].*

- b. During QR, a summary of tests reports and software user manual are reviewed. The consistency of all software documentation (RB, TS, DDF, DJF) shall be verified.

EXPECTED OUTPUT: *Customer’s approval of qualified state [DJF; QR].*

#### **5.3.2.13 Acceptance review**

After the qualification review, the customer shall hold an acceptance review (AR).

AIM: Acceptance of the software with respect to the intended operational environment.

EXPECTED OUTPUT: *Customer’s approval of accepted state [DJF; AR].*

#### **5.3.2.14 Validation activities phasing with regard to the acceptance review**

The validation with respect to the requirements baseline shall be performed not later than the acceptance review

EXPECTED OUTPUT: *Phasing of activities of the software validation with respect to the requirements baseline in the software development plan [MGT; SRR, PDR].*

#### **5.3.2.15 Software procurement process implementation**

The supplier shall implement the software procurement process for reused software, and document the process in the software development plan, as described in the ECSS-Q-80B subclause 5.7.1.

EXPECTED OUTPUT: *Software procurement process implementation in the software development plan [MGT; SRR, PDR].*

### **5.3.3 Joint technical review process**

#### **5.3.3.1 Introduction**

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

#### **5.3.3.2 Support to software reviews**

- a. The software support to joint technical reviews shall be related to project phasing and planning (refer to ECSS-M-30)..
- b. software shall undergo the overall software milestone reviews SRR, PDR, DDR, CDR, QR and AR as a minimum
- c. The supplier shall support any further reviews requested by the customer.

NOTE Internal reviews can be replaced by inspections.

EXPECTED OUTPUT: *Milestone review reports [DJF; SRR, PDR, DDR, CDR, QR, AR].*

#### **5.3.3.3 Technical reviews**

- a. Technical reviews (including milestone reviews) shall be held to evaluate the software products or services under consideration and provide evidence that:
  - they are complete;
  - they conform to applicable standards and specifications;
  - changes are properly implemented and affect only those areas identified by the configuration management process;
  - they adhere to applicable schedules;
  - they are ready for the next activity;
  - the development, operation, or maintenance is being conducted according to the plans, schedules, standards, and guidelines laid down for the project.

EXPECTED OUTPUT: *Technical review reports [DJF; SRR, PDR, DDR, CDR, QR, AR].*

- b. Reviews of each identified software product shall be planned within its defined software life cycle according to the criteria above.

EXPECTED OUTPUT: *Technical review reports [DJF; SRR, PDR, DDR, CDR, QR, AR].*

### **5.3.4 Interface management**

#### **5.3.4.1 Interface definition**

Interfaces shall be defined in the requirements baseline in an interface requirements document, which defines the requirements applicable to various elements of the system product tree.

EXPECTED OUTPUT: *Interface requirements document [IRD(RB); SRR].*

#### 5.3.4.2 Interface management procedures

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

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

EXPECTED OUTPUT: *a. Interface management procedures [RB; SRR];  
b. Part of configuration management requirements [RB; SRR].*

### 5.3.5 Technical budget and margin management

#### 5.3.5.1 Software technical budget and margin philosophy definition

Technical budget targets and margin philosophy dedicated to the software shall be specified by the customer in the requirements baseline.

AIM: To define the limits of software budgets associated with computer resources (such as: CPU load and maximum memory size) and performance requirements to be considered by the supplier.

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

#### 5.3.5.2 Software technical budget management

The supplier shall manage margins regarding the technical budgets and present their status at each milestone, describing the utilized analytical hypothesis.

AIM: To establish the margins by analysis in the early development and to consolidate them by performance measurements commensurate with the software implementation.

EXPECTED OUTPUT: *Margins and technical budgets status [DJF; PDR, DDR, CDR, QR, AR].*

## 5.4 Software requirements and architecture engineering process

### 5.4.1 Introduction

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

- software requirements analysis;
- software architectural design.

### 5.4.2 Software requirements analysis

#### 5.4.2.1 Establishment and documentation of software requirements

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

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

- a. Functional and performance specifications, including hardware characteristics, and environmental conditions under which the software item executes, including budgets requirements [TS; PDR];*
- b. Software product quality requirements (see ECSS-Q-80B subclause 7.2) [TS; PDR];*
- c. Security specifications, including those related to factors which can compromise sensitive information [TS; PDR];*

- d. *Human factors engineering (ergonomics) specifications, including those related to manual operations, human-equipment interactions, constraints on personnel, and areas requiring concentrated human attention, that are sensitive to human errors and training [TS; PDR];*
- e. *Data definition and database requirements [TS; PDR];*

EXPECTED OUTPUT: *Interface control document [TS; PDR]*

- f. *Interfaces external to the software item [ICD(TS); PDR].*

#### **5.4.2.2 Definition of functional and performance requirements for in flight modification**

When in flight modification is specified for flight software, the supplier shall perform analysis of the specific implications for the software design and validation processes and include the functional and performance requirements in the technical specification and the corresponding design in the software architectural design.

EXPECTED OUTPUT: a. *Specifications for in flight software modifications [TS; PDR];*

- b. *Design for in flight modification [DDF; PDR].*

#### **5.4.2.3 Identification of requirement unique identifier**

Each requirement shall be separately identified in order to allow for traceability.

EXPECTED OUTPUT: *Requirements unique identifier [TS; PDR].*

#### **5.4.2.4 Definition of a software logical model**

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

NOTE 1 The logical model can be the result of an iterative verification process with the customer. It also supports the requirements capture, documents and formalizes the software requirements.

NOTE 2 A logical model is a representation of the technical specification, independent of the implementation, written with a formalized language and it can be possibly executable. Formal methods can be used to prove properties of the logical model itself and therefore of the technical specification. The logical model allows in particular to verify that a technical specification is complete (i.e. by checking a software requirement exists for each logical model element), and consistent (because of the model checking).  
The logical model can be completed by specific feasibility analyses such as benchmarks, in order to check the technical budgets (e.g. memory size and computer throughput). In case the modelling technique allows for it, preliminary automatic code generation can be used to define the contents of the software validation test specification.

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

#### **5.4.2.5 Definition of behavioural view**

The logical model shall include a behavioural view.

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

#### 5.4.2.6 MMI software mock-up development

- a. The supplier shall develop a software mock-up to support the requirements and architecture engineering process, in accordance with customer's requirements.

EXPECTED OUTPUT: *MMI specifications for software [TS; PDR].*

- b. The supplier shall use the mock-up to prototype the specifications of man-machine interfaces for the software, such that MMI specifications are consolidated and evaluated with respect to human factors and use.

AIM: The aim of this subclause includes:

- proper consideration of human factors,
- that the man-machine interface achieves an acceptable state of definition during requirements and architecture engineering activities, and
- that the technical performance of the man-machine interface is verified.

NOTE Depending on the nature of the project, the supplier can opt to later upgrade the software mock-up of the MMI to become part of the final software product. However, unless the mock-up is later upgraded to become part of the final product tree, the mock-up need not be a formal delivery to the customer.

EXPECTED OUTPUT: *Report on evaluation of MMI specifications using a software mock-up [DJF; PDR].*

- c. The customer shall ensure that end-users, or their representatives, participate in the MMI mock-up evaluation.

### 5.4.3 Software architectural design

#### 5.4.3.1 Transformation of software requirements into a software architecture

The supplier shall transform the requirements for the software item into an architecture that describes its top-level structure and identifies the software components, ensuring that all the requirements for the software item are allocated to its software components and later refined to facilitate detailed design.

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

#### 5.4.3.2 Software architectural design description

The architectural design description shall as a minimum cover hierarchy, dependency, interfaces and operational usage for the software components.

EXPECTED OUTPUT: *Hierarchy, dependency and interfaces of software components in the software architectural design [DDF; PDR]*

#### 5.4.3.3 Software architectural design documentation

The architectural design description shall document the process, data and control aspects of the product.

EXPECTED OUTPUT: *Process, data and control aspects of software components in the software architectural design [DDF; PDR].*

#### 5.4.3.4 Software architectural design contents

The software architectural design shall describe:

- the static architecture (i.e. decomposition into software elements such as packages and classes or modules),
- the dynamic architecture, which involves active objects such as threads, tasks and processes,

- the mapping between the static and the dynamic architecture, and
- the software behaviour.

EXPECTED OUTPUT: a. *Software static architecture [DDF; PDR];*  
b. *Software dynamic architecture [DDF; PDR];*  
c. *Software behaviour [DDF; PDR].*

#### **5.4.3.5 Software design method**

The supplier shall use a method (e.g. object oriented or functional) to produce the static architecture including:

- software elements, their interfaces, and;
- software elements relationships

#### **5.4.3.6 Selection of a computational model for real-time software**

- a. The dynamic architecture design shall select an analysable computational model and shall be described accordingly.

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

- b. Scheduling simulations shall be performed.

EXPECTED OUTPUT: *Scheduling simulation report [DJF; PDR].*

#### **5.4.3.7 Description of software dynamic behaviour**

The software architecture design shall also describe the dynamic behaviour of the software, for instance by means of description techniques based on automata and scenarios or techniques used for the behavioural view of the logical model (see subclause 5.4.2.5).

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

#### **5.4.3.8 Development and documentation of the software interfaces**

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

EXPECTED OUTPUT: a. *Preliminary external interfaces design [ICD(TS); PDR];*  
b. *Preliminary internal interfaces design [DDF; PDR].*

#### **5.4.3.9 Definition of methods and tools for software to be reused**

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

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

#### **5.4.3.10 Evaluation of potential reuse of software**

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

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

#### **5.4.3.11 Evaluation of reuse of predeveloped software**

The supplier shall consider the “reuse” of already developed, commercial off-the-shelf, modifiable off-the-shelf software, free software and open source (see also subclause 6.2.7 in ECSS-Q-80B).

EXPECTED OUTPUT: *Specification of reuse of predeveloped software [TS; PDR].*

#### 5.4.3.12 Analysis of potential reusability

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

- identification of the reuse components with respect to the functional requirements baseline, and
- a quality evaluation of these components, invoking ECSS-Q-80B subclause 6.2.6.

NOTE There are no special requirements concerning the verification and validation requirements for reused software. The requirements are the same as for software developed without reuse. The difference is that some existing verification and validation plans and results can be available with the reused products. However, the full verification and validation requirements apply to reused software as for any other part of the software development.

EXPECTED OUTPUT: *Justification of reuse with respect to requirements baseline in the software reuse file [DJF; PDR].*

#### 5.4.3.13 Definition and documentation of the software integration requirements and plan

The supplier shall define and document preliminary test requirements and the plan for software integration.

EXPECTED OUTPUT: *Software integration test plan (preliminary) [DJF; PDR].*

#### 5.4.3.14 Conducting a preliminary design review

The supplier shall conduct a preliminary design review (PDR) in accordance with subclause 5.3.2.8.

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

EXPECTED OUTPUT: *PDR milestone report [DJF; PDR].*

## 5.5 Software design and implementation engineering process

### 5.5.1 Introduction

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

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

NOTE The design of software items activity includes only a review at DDR of all the subclauses' expected outputs in the case of flight software

### 5.5.2 Design of software items

#### 5.5.2.1 Detailed design of each software component

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

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

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

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



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

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

#### **5.5.2.2 Development and documentation of the software interfaces detailed design**

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

EXPECTED OUTPUT: a. *External interfaces design (update) [ICD(TS); DDR, CDR];*

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

#### **5.5.2.3 Production of software items physical model**

The software design shall produce the physical model of the software items described during the software architectural design.

NOTE The physical model includes the static design, the dynamic design, the mapping between the static and the dynamic views, and the behaviour of the software elements.

EXPECTED OUTPUT: a. *Software static design [DDF; DDR, CDR];*

b. *Software dynamic design [DDF; DDR, CDR];*

c. *Software elements behaviour [DDF; DDR, CDR];*

#### **5.5.2.4 Utilization of methods for software static design**

A design method (e.g. object oriented or functional method) shall be used to produce the static design including:

- software elements, their interfaces, and
- software elements relationships.

#### **5.5.2.5 Description of the dynamic aspects of physical model for real-time software**

- a. The dynamic design shall be based on the computational model selected during the software architectural design and shall describe the dynamic aspect of the physical model accordingly.

EXPECTED OUTPUT: *Software dynamic design [DDF; DDR, CDR].*

- b. Scheduling simulations shall be performed.

EXPECTED OUTPUT: *Scheduling simulation report [DJF; DDR, CDR].*

#### **5.5.2.6 Utilization of description techniques for the software behaviour**

The software design shall also describe the dynamic behaviour of the software elements, for instance by means of description techniques based on automata and scenarios or techniques used for the behavioural view of the logical model (see subclause 5.4.2.5)

EXPECTED OUTPUT: *Dynamic behaviour [DDF; DDR, CDR].*

#### **5.5.2.7 Determination of design method consistency for real-time software**

In some cases, several design methods can be used for different items of the same software. In this case, it shall be ensured that all the utilized methods are, from a dynamic stand-point, consistent among themselves and consistent with the selected computational model.

EXPECTED OUTPUT: *Compatibility of design methods with the computational model [DDF; DDR, CDR].*



**5.5.2.8 Development and documentation of the software user manual**

The supplier shall develop and document the software user manual.

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

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

The supplier shall define and document test requirements and plan for testing software units, including stressing the software at the limits of its requirements.

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

**5.5.2.10 Updating of the software integration test requirements and plan**

The supplier shall update the test requirements and the plan for software integration.

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

**5.5.2.11 Conducting a detailed design review for flight software**

The supplier shall conduct a detailed design review (DDR) in accordance with subclause 5.3.2.9

EXPECTED OUTPUT: *DDR milestone report [DJF; DDR].*

**5.5.3 Coding and testing****5.5.3.1 Development and documentation of the software units, test procedures and test data**

The supplier shall develop and document the following:

- the coding of each software unit;
- the build procedures to compile and link software units;
- test procedures and data for testing each software unit.

EXPECTED OUTPUT: *a. Software component design documents and code (update) [DDF; CDR];*

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

*c. Software unit test plan (update) [DJF; CDR].*

**5.5.3.2 Software unit testing**

The supplier shall test each software unit ensuring that it satisfies its requirements and document the test results

EXPECTED OUTPUT: *a. Software component design document and code (update) [DDF; CDR];*

*b. Software unit test reports [DJF; CDR].*

**5.5.3.3 Software user manual updating**

The supplier shall update the software user manual.

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

**5.5.3.4 Updating of the software integration test requirements and plan**

The supplier shall update the test requirements and the plan for software integration.

AIM: To make the test requirements and integration plan consistent with the results of the code design process.

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

## 5.5.4 Integration

### 5.5.4.1 Software integration test plan development

The supplier shall develop and document an integration plan to integrate the software units and software components into the software item, providing the following data:

- test requirements;
- test procedures;
- test data;
- responsibilities allocation;
- schedule information.

EXPECTED OUTPUT: *Software integration test plan [DJF; CDR].*

### 5.5.4.2 Software units and software component integration and testing

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

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

### 5.5.4.3 Software user manual updating

The supplier shall update the software user manual.

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

## 5.6 Software validation process

### 5.6.1 Introduction

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

The software validation process consists of:

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

### 5.6.2 Validation process implementation

#### 5.6.2.1 Determination of the validation effort for the project

The validation effort and the degree of organizational independence of that effort shall be determined, coherent with ECSS-Q-80B subclause 6.3.4.2.

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

#### 5.6.2.2 Establishment of a validation process

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

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

- b. Validation tasks defined in subclauses 5.6.3 and 5.6.4, including associated methods, techniques, and tools for performing the tasks, shall be selected.

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

#### **5.6.2.3 Selection of an ISVV organization**

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

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

- b. The conductor shall be assured of the independence and authority to perform the validation tasks.

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

NOTE 1 This subclause is applied with ECSS-M-20 and ECSS-Q-80B, subclause 6.3.4.29.

NOTE 2 The conductor is the person or the entity that takes in charge the validation tasks (e.g. test cases specification, design, execution and management)

#### **5.6.2.4 Development and documentation of a validation plan**

A validation plan shall be developed and documented, including, as a minimum the following:

- items subject to validation;
- validation tasks to be performed;
- resources, responsibilities, and schedule for validation;
- procedures for forwarding validation reports to the customer and other parties.

EXPECTED OUTPUT: *Software validation plan - organization and activities [DJF; PDR].*

### **5.6.3 Validation activities with respect to the technical specification**

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

The supplier shall develop and document, for each validation requirement of the software item, a set of tests, test cases (inputs, outputs, test criteria) and test procedures for conducting software validation testing.

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

#### **5.6.3.2 Conducting the validation with respect to the technical specification**

- a. The validation tests shall be conducted as specified in the output of subclause 5.6.3.1 above, including:
- testing with stress, boundary, and singular inputs;
  - testing the software product for its ability to isolate and minimize the effect of errors; that is graceful degradation upon failure, request for operator assistance upon stress, boundary and singular conditions;
  - testing that the software product can perform successfully in a representative operational environment.

EXPECTED OUTPUT: *Validation testing report with respect to the technical specification [DJF; CDR].*

- b. Priority shall be given to validation testing. If not possible, other validation methods such as analysis, inspection or review of design shall be used with an appropriate justification.

EXPECTED OUTPUT: *Verification report [DJF; CDR]*

#### **5.6.3.3 Updating the software user manual**

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

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

#### **5.6.3.4 Test readiness review**

Test readiness reviews shall be held before the commencement of key test activities.

EXPECTED OUTPUT: *Technical review report [DJF; TRR].*

#### **5.6.3.5 Conducting a critical design review**

The supplier shall conduct a critical design review (CDR) in accordance with subclause 5.3.2.10.

EXPECTED OUTPUT: *CDR milestone report [DJF; CDR].*

### **5.6.4 Validation activities with respect to the requirements baseline**

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

The supplier shall develop and document, for each validation requirement of the software item, a set of tests, test cases (inputs, outputs, test criteria) and test procedures for conducting software validation testing.

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

#### **5.6.4.2 Conducting the validation with respect to the requirements baseline**

- a. The validation tests shall be conducted as specified in the output of subclause 5.6.4.1 above, including:
- testing with stress, boundary, and singular inputs;
  - testing the software product for its ability to isolate and minimize the effect of errors; that is graceful degradation upon failure, request for operator assistance upon stress, boundary and singular conditions;
  - testing that the software product can perform successfully in a representative operational environment.

EXPECTED OUTPUT: *Validation testing report with respect to the requirements baseline [DJF; QR, AR].*

- b. Priority shall be given to validation testing. If not possible, other validation methods such as analysis, inspection or review of design shall be used with an appropriate justification.

EXPECTED OUTPUT: *Verification report [DJF; QR, AR]*

#### **5.6.4.3 Updating the software user manual**

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

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

#### **5.6.4.4 Test readiness review**

Test readiness reviews shall be held before the commencement of key test activities.

EXPECTED OUTPUT: *Technical review report [DJF; TRR].*

#### **5.6.4.5 Conducting a qualification review**

The qualification review (QR) shall be conducted in accordance with subclause 5.3.2.12.

AIM: To verify that the software meets all the requirements, and in particular that verification and validation process outputs enable transition to “qualified state” for the software products.

EXPECTED OUTPUT: *a. Software release document [DDF; QR];  
b. Software delivery [DDF; QR];  
c. Software design and test evaluation report [DJF; QR];  
d. Validation testing report [DJF; QR];  
e. Test specification evaluation [DJF; QR];  
f. QR milestone report [DJF; QR].*

## **5.7 Software delivery and acceptance process**

### **5.7.1 Introduction**

This process consists of the following activities:

- software delivery and installation;
- software acceptance.

### **5.7.2 Software delivery and installation**

#### **5.7.2.1 Preparation of the software product**

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

EXPECTED OUTPUT: *a. Software delivery [DDF; QR, AR];  
b. Software release document [DDF; QR, AR].*

#### **5.7.2.2 Supplier's provision of training and support**

The supplier shall provide initial and continuing training and support to the customer as specified in the requirement baseline.

EXPECTED OUTPUT: *Training material [DDF; QR].*

#### **5.7.2.3 Installation planning**

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

EXPECTED OUTPUT: *Installation plan [DJF; AR].*

#### **5.7.2.4 Installation activities reporting**

- a. The resources and information to install the software product shall be determined and be available.
- b. The supplier shall assist the customer with the set-up activities.
- c. It shall be ensured that the software code and databases initialize, execute and terminate as specified in the installation plan.
- d. The installation events and results shall be documented.

EXPECTED OUTPUT: *Installation report [DJF; AR].*

### **5.7.3 Software acceptance**

#### **5.7.3.1 Acceptance test planning**

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

EXPECTED OUTPUT: *Acceptance test plan [DJF; AR]*

#### **5.7.3.2 Acceptance test execution**

The customer shall perform the acceptance testing.

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

#### **5.7.3.3 Executable code generation and installation**

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

EXPECTED OUTPUT: *Executable code generation test in the acceptance test plan [DJF; AR].*

#### **5.7.3.4 Supplier's support to customer's acceptance**

- a. The supplier shall support the customer's acceptance reviews and testing of the software product.

EXPECTED OUTPUT: *AR milestone report [DJF; AR].*

- b. Acceptance reviews and testing shall consider the results of the joint reviews (ECSS-Q-20B subclause 6.6.4.4 and 10.3), audits (ECSS-Q-20B subclause 4.6), testing and validation (ECSS-Q-80B subclause 6.3.4), and system validation testing (if performed).

EXPECTED OUTPUT: *AR milestone report [DJF; AR].*

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

EXPECTED OUTPUT: *Acceptance testing documentation [DJF; AR].*

#### **5.7.3.5 Evaluation of acceptance testing**

The acceptance tests shall be evaluated with respect to the requirements baseline.

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

#### **5.7.3.6 Conducting an acceptance review**

- a. The acceptance review (AR) shall be conducted in accordance with subclause 5.3.2.13.

EXPECTED OUTPUT: *AR milestone report [DJF; AR].*

- b. The software supplier's acceptance support task (see subclause 5.7.3.4) shall support the customer's acceptance activities in preparation of the AR.

EXPECTED OUTPUT: *a. Software acceptance data package [DJF; AR];  
b. Acceptance testing documentation [DJF; AR];  
c. Software release document [DDF; AR];  
d. Software delivery [DDF; AR].*

## 5.8 Software verification process

### 5.8.1 Introduction

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

**NOTE** The technical specification is accompanied by the planning of verification and validation of the software product. These plannings are determined by the customer's requirements baseline (see subclause 5.2.4.2) and by invoking the relevant verification and validation processes (see subclauses 5.6.2. and 5.8.2)

The software verification process consists of:

- verification process implementation, and
- verification activities.

### 5.8.2 Verification process implementation

#### 5.8.2.1 Determination of the verification effort for the project

- a. A determination shall be made concerning the verification effort and the degree of organizational independence.

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

- b. Applicability of ECSS-M-00B subclause 5.3 (management of risks), and ECSS-Q-80B subclauses 6.2.2 (software dependability and safety) and 6.2.6.13 (independent software verification and validation) shall be checked.

**EXPECTED OUTPUT:** *Software verification plan - identification of risks and level of independence [DJF; PDR].*

- c. The project requirements shall be analysed in terms of:
  - the potential for an undetected error in a system or software requirement to cause death or personal injury, mission failure, or financial or catastrophic equipment loss or damage;
  - the maturity of and risks associated with the software technology to be used;
  - availability of funds and resources.

**EXPECTED OUTPUT:** *Software verification plan - criticality and resources identification [DJF; PDR].*

#### 5.8.2.2 Establishment of the verification process, methods and tools

- a. A verification process shall be established to verify the software products.

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

- b. Target life cycle activities and software products needing verification shall be determined based upon the scope, magnitude, complexity, and criticality analysis mentioned in 5.8.2.1

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



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

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

#### **5.8.2.3 Selection of the organization responsible for conducting the verification**

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

EXPECTED OUTPUT: *Independent verification organization selection [DJF; PDR].*

- b. This organization shall have the independence and authority needed to perform the verification activities.

NOTE ECSS-Q-80B subclause 6.2.6.13 (independent software verification) and ECSS-M-20 (project organization) contain further requirements relevant for this subclause.

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

EXPECTED OUTPUT: *Appropriate element of project requirements documents dealing with project organization [DJF; PDR].*

#### **5.8.2.4 Development and documentation of a verification plan covering the software verification activities**

Based upon the verification tasks as determined, a verification plan shall be developed and documented, addressing the following items:

- the life cycle activities and software products subject to verification;
- the required verification tasks for each life cycle activity, software product, related resources, responsibilities, and schedule;
- the procedures for forwarding verification reports to the customer and other involved organizations.

EXPECTED OUTPUT: *Software verification plan - organization and activities [DJF; PDR].*

### **5.8.3 Verification activities**

#### **5.8.3.1 Verification of software requirements**

The supplier shall verify the software requirements considering the criteria listed below:

- software requirements are traceable to system partitioning and system requirements;
- software requirements are externally and internally consistent (not implying formal proof consistency);
- software requirements are verifiable;
- feasibility of software design;
- feasibility of operations and maintenance;
- the software requirements related to safety, security, and criticality are correct as shown by suitably rigorous methods.

EXPECTED OUTPUT: *a. Requirements traceability matrices [DJF; PDR];*

*b. Requirements verification report [DJF; PDR].*

#### **5.8.3.2 Verification of the software architectural design**

The supplier shall verify the architecture of the software item and the interface design considering the criteria listed below:



- external consistency with the requirements of the software item;
- internal consistency between the software components;
- traceability from the requirements to the software item;
- feasibility of producing a detailed design;
- feasibility of operations and maintenance;
- the design is correct with respect to the requirements and the interfaces;
- the design implements proper sequence of events, inputs, outputs, interfaces, logic flow, allocation of timing and sizing budgets, and error handling;
- the chosen design can be derived from requirements;
- the design implements safety, security and other critical requirements correctly as shown by suitable rigorous methods.

EXPECTED OUTPUT: a. *Software architectural design to requirements traceability matrices [DJF; PDR];*  
b. *Software architectural design and interface verification report [DJF; PDR].*

#### 5.8.3.3 Verification of the software detailed design

The supplier shall verify the software detailed design and test requirements in accordance with the criteria listed below:

- traceability to the architectural design of the software item;
- external consistency with architectural design;
- internal consistency between software components and software units;
- feasibility of testing;
- feasibility of operation and maintenance;
- the design is correct with respect to requirements and interfaces;
- the design implements proper sequence of events, inputs, outputs, interfaces, logic flow, allocation of timing and sizing budgets, and error handling;
- the chosen design can be derived from requirements;
- the design implements safety, security, and other critical requirements correctly as shown by suitable rigorous methods.

EXPECTED OUTPUT: a. *Detailed design traceability matrices [DJF; CDR];*  
b. *Detailed design verification report [DJF; CDR].*

#### 5.8.3.4 Verification of code

The supplier shall verify the software code and tests results considering the criteria listed below:

- the code is traceable to design and requirements, testable, correct, and in conformity to software requirements and coding standards;
- the code implements proper event sequence, consistent interfaces, correct data and control flow, completeness, appropriate allocation timing and sizing budgets, and error handling;
- the chosen code can be derived from design or software requirements;
- the code implements safety, security, and other critical requirements correctly as shown by suitable rigorous methods;
- external consistency with the requirements and design of the software item;
- internal consistency between software units;
- absence of run-time errors;
- test coverage of units;

- feasibility of software integration and testing;
- feasibility of operation and maintenance.

EXPECTED OUTPUT: a. *Software code traceability matrices [DJF; CDR];*  
b. *Software code verification report [DJF; CDR].*

#### **5.8.3.5 Verification of software integration**

- a. The supplier shall verify the software integration considering that the software components and units of each software item are completely and correctly integrated into the software item.

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

- b. In addition, the supplier shall verify the software integration test plan, design, code, tests, test results and software user manual, considering the criteria specified below:
- external consistency with system requirements;
  - traceability to software architectural design;
  - internal consistency;
  - interface testing coverage;
  - requirements test coverage;
  - conformance to expected results;
  - feasibility of software validation testing;
  - feasibility of operation and maintenance.

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

#### **5.8.3.6 Verification of software documentation**

The supplier shall verify the software documentation considering the criteria listed below:

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

AIM: That the supplier baselines the documentation for the project to transit from “specified state” to the “defined state”, thereby achieving the milestone of a completed design.

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

#### **5.8.3.7 Verification of test specifications**

The supplier shall demonstrate that the test requirements, test cases, and test specifications cover all software requirements of the technical specification or the requirements baseline.

EXPECTED OUTPUT: a. *Traceability of the requirements baseline to the validation tests [DJF; QR, AR];*  
b. *Traceability of the technical specification to the validation tests [DJF; CDR].*

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

The supplier shall verify the design, code, tests, test results and software user manual in accordance with the criteria listed below:

- test coverage of the requirements of the software item;
- conformance to expected results;

- feasibility of system integration and testing, if conducted;
- feasibility of operation and maintenance.

EXPECTED OUTPUT: a. *Validation report evaluation with respect to the technical specification [DJF; CDR];*  
b. *Validation report evaluation with respect to the requirements baseline [DJF; QR, AR].*

#### 5.8.3.9 Evaluation of validation: complementary system level validation

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

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

EXPECTED OUTPUT: *Complement of validation at system level [DJF; AR].*

#### 5.8.3.10 Problem and nonconformance handling

- a. Problems and nonconformances detected by the software verification, the software validation and the CDR shall be entered into the problem resolution process (ECSS-Q-80B subclauses 5.4.5 and 5.4.6).

EXPECTED OUTPUT: *Problem and nonconformance reports [DJF; CDR, QR, AR].*

- b. Results of the verification activities shall be made available to the customer and other involved organizations.

EXPECTED OUTPUT: *Problem and nonconformance reports [DJF; CDR, QR, AR].*

#### 5.8.3.11 Schedulability analysis for real-time software

- a. As support to the verification of the software requirements and architectural design (feasibility of the design), the supplier shall use an analytical model to perform a schedulability analysis and prove that the design is feasible. If no analytical model exists, modelling or simulation shall be used to prove the feasibility of the design.

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

EXPECTED OUTPUT: *Schedulability analysis or design feasibility verification with models or simulation [DJF; PDR].*

- b. As support to the verification of the software detailed design (feasibility of operations), the supplier shall refine the schedulability analysis performed during the software architectural design on the basis of the software detailed design documentation

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

- c. As support to the verification of the software coding and testing (feasibility of operations), the supplier shall refine the schedulability analysis performed during the software detailed design with the actual information extracted from the code.

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

#### 5.8.3.12 Technical budgets management

- a. As support to the verification of the software requirements and architectural design (feasibility of the design), the supplier shall perform an estimation of the technical budgets:

- The memory size for static code size, static data size and stack size;
- The CPU utilization.

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

- b. As support to the verification of the software detailed design (feasibility of operations), the supplier shall refine the estimation of the technical budgets:
- The memory size refined for static code size, static data size and stack size expressed on a thread basis, measuring them per lowest level design component;
  - The CPU utilization, refined, considering the worst case execution time of each lowest level design component having its own control flow (therefore including the call to the protected objects).

NOTE The worst case execution time of each lowest level design component having its own control flow is multiplied by the number of times this component is executed per second. The resulting quantity is summed over all other design components. The result is the estimated percentage processor utilization.

EXPECTED OUTPUT: *Technical budgets - memory and CPU estimation [DJF; DDR, CDR].*

- c. As support to the verification of the coding and testing (feasibility of operations), the technical budgets shall be updated with the measured values and shall be compared to the margins.

EXPECTED OUTPUT: *Technical budgets (update) - memory and CPU calculation [DJF; CDR].*

#### **5.8.3.13 Behaviour modelling verification**

- a. As support to the verification of the software requirements and architectural design, the supplier shall verify the software behaviour using the behavioural view of the logical model produced in subclause 5.4.2.5.

EXPECTED OUTPUT: *Software behaviour verification [DJF; PDR].*

- b. As support to the verification of the software detailed design, the supplier shall model the software behaviour and verify by means of the techniques used for the description of the software behaviour

EXPECTED OUTPUT: *Software behaviour verification [DJF; DDR, CDR].*

#### **5.8.3.14 Verification of design: feasibility of testing**

The supplier shall verify the feasibility of testing under the following aspects:

- Appropriate verification points are identified and included in the detailed design in order to prepare the effective testing of the performance requirements.
- Assertions defining computational invariant properties, or temporal properties (possibly derived from the behavioural model) are added within the design.
- Capability of fault injection.

EXPECTED OUTPUT: *Testing feasibility report [DJF; CDR].*

## **5.9 Software operation process**

### **5.9.1 Introduction**

The operation process may start after completion of software acceptance. Since software products are an integrated part of the space system, the phasing and

management of operation is determined by the overall system requirements and applied to the software products. The operation process is therefore not directly connected to the overall mission phase E, but is determined by the system level requirement to operate the software product at a given time. Ground segment software products for example are in extensive operational use to qualify the ground segment, well before the actual mission operations occur. Similarly, for flight software, extensive ground operations are, in general, performed for testing flight equipment long before space system flight operations begin.

Both the documents and the reviews identified as outputs by the subclauses of 5.9 are therefore part of the operations activities for the space systems, and the requirements for these reviews and their documentation form part of the space system operations requirements covered in other ECSS standards. The provisions of this subclause 5.9 are intended to produce the required software engineering inputs for the system level activities.

The operation process comprises the activities and tasks of the operator. The process covers the operation of the software product and operational support to users. Because operation of a software product is integrated into the operation of the system, the activities and tasks of this process refer to the system.

The operator manages the operation process at the project level following the management process (ECSS-M-30). This process consists of the following activities:

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

NOTE 1 In ECSS-E-40, the operator administrates the software product, supports the user in the resolution of problems (e.g. helpdesk function), links with the maintainer for the evolutions and deploys the new versions of the software product.

NOTE 2 For a satellite, in the meaning of ECSS-E-40, using flight software consists of e.g. sending telecommands whilst operating the flight software consists of e.g. patching or rebooting the software.

## 5.9.2 Process implementation

### 5.9.2.1 Operational plans and standards development

The operator shall develop, document and execute a plan and set operational standards for performing the activities and tasks of this process.

EXPECTED OUTPUT: *Operational plan - plan and standards [OP; ORR].*

### 5.9.2.2 Problem handling procedures definition

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

EXPECTED OUTPUT: *Operational plan - procedures for problem handling [OP; ORR].*

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

EXPECTED OUTPUT: *Problem and nonconformance report [OP].*

NOTE ECSS-Q-80B subclause 5.4.5 (nonconformances) and subclause 5.4.6 (software problems) contain further requirements relevant for this subclause.

### 5.9.2.3 Operational testing definition

The operator shall establish procedures for:

- testing the software product in its operation environment,
- entering problem reports and modification requests to the maintenance process (see subclause 5.10), and
- releasing the software product for operational use in accordance with the change control established and maintained in conformance with ECSS-M-40.

EXPECTED OUTPUT: *Operational plan - operational testing specifications [OP; ORR].*

## 5.9.3 Operational testing

### 5.9.3.1 Operational testing execution

- a. For each release of the software product, the operator shall perform operational testing in line with the change control established and maintained in conformance with ECSS-M-40.

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

- b. On satisfying the specified criteria, the software product shall be released for operational use.

EXPECTED OUTPUT: *Software delivery [OP; ORR].*

### 5.9.3.2 Software operational requirements demonstration

- a. The customer shall ensure that prior to the operations, the software has been demonstrated capable of implementing the operational requirements.

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

EXPECTED OUTPUT: *Validation of operational requirements [OP; ORR].*

- b. This demonstration as specified in a. shall be representative in terms of:
  - hardware operating environment,
  - situations to which the software is designed to be fault tolerant,
  - system configuration,
  - sequence of operations, and
  - operator interventions.

EXPECTED OUTPUT: *Demonstration criteria [OP; ORR].*

## 5.9.4 Software operation

The software shall be operated in its intended environment according to the software user manual.

## 5.9.5 User support

### 5.9.5.1 User's assistance

- a. The operator shall provide assistance and consultation to the users.
- b. User's requests and subsequent actions shall be recorded and monitored.

EXPECTED OUTPUT: *Record of the user's request and subsequent actions [OP].*

**5.9.5.2 Handling of user's requests**

- a. The operator shall forward user requests to the maintenance process for resolution.
- b. User's requests shall be addressed and the actions that are planned and taken shall be reported to the originators of the requests.

**5.9.5.3 Provisions of work-around solutions**

- a. If a reported problem has a temporary work-around solution before a permanent solution can be released, the originator of the problem report shall be given the option to use it.
- b. Permanent corrections, releases that include previously omitted functions or features, and system improvements shall be applied to the operational software product using the maintenance process (subclause 5.10).

**5.10 Software maintenance process****5.10.1 Introduction**

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

The activities provided in this subclause 5.10 are specific to the maintenance process; however, the process can utilize other processes in this Standard. If the software engineering processes (subclause 4.2) are utilized, the term supplier there is interpreted as maintainer.

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

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

This process consists of the following activities:

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

**5.10.2 Process implementation****5.10.2.1 Software maintenance process planning**

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

EXPECTED OUTPUT: *Maintenance plan - plans and procedures [MF; QR].*

**5.10.2.2 Software maintenance process procedures, methods and standards**

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

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



#### 5.10.2.3 Problem reporting and handling

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

EXPECTED OUTPUT: *Maintenance plan - problem reporting and handling [MF; QR].*

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

NOTE ECSS-Q-80B subclause 5.4.5 (nonconformances) and subclause 5.4.6 (software problems) contain further requirements relevant for this subclause.

EXPECTED OUTPUT: *Problem and nonconformance report [MF].*

#### 5.10.2.4 Implementation of configuration management process

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

EXPECTED OUTPUT: *Maintenance plan - configuration management process [MF; QR].*

#### 5.10.2.5 Long term maintenance for flight software

- a. The maintenance plan shall take into account the spacecraft lifetime.

EXPECTED OUTPUT: *Maintenance plan [MF; QR].*

- b. If this lifetime goes after the expected obsolescence date of the software engineering environment, then the maintainer shall propose solutions to be able to produce and upload modifications to the spacecraft up to its end of life.

EXPECTED OUTPUT: *Long term maintenance solutions [MF; QR].*

### 5.10.3 Problem and modification analysis

#### 5.10.3.1 Problem analysis

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

- type (e.g. corrective, improvement, preventive, or adaptive to new environment);
- scope (e.g. size of modification, cost involved, and time to modify);
- criticality (e.g. impact on performance, safety, or security).

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

#### 5.10.3.2 Problem verification

The maintainer shall reproduce or verify the problem.

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

#### 5.10.3.3 Development of options for modifications

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

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



#### **5.10.3.4 Documentation of problem, analysis and implementation**

The maintainer shall document the problem or the modification request, the analysis results and implementation options in the problem analysis report or in the modification analysis report, respectively.

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

#### **5.10.3.5 Customer approval of selected modification options**

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

### **5.10.4 Modification implementation**

#### **5.10.4.1 Analysis and documentation of product modification**

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

EXPECTED OUTPUT: *Modification identification [MF].*

#### **5.10.4.2 Documentation of software product changes**

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

#### **5.10.4.3 Invoking of software engineering processes for modification implementation**

The maintainer shall enter the software engineering processes (subclauses 5.3, 5.4, 5.5, 5.6, 5.7, 5.8) to implement the modifications and consider the following:

- Test and evaluation criteria for testing and evaluating the modified and the unmodified parts (software units, components, and configuration items) of the system shall be defined and documented.
- The complete and correct implementation of the new and modified requirements shall be ensured.
- It also shall be ensured that the original, unmodified requirements have not been affected.
- The test results shall be documented.

### **5.10.5 Conducting maintenance reviews**

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

EXPECTED OUTPUT: *Baseline for changes [MF].*

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

EXPECTED OUTPUT: *Baseline for changes [MF].*

### **5.10.6 Software migration**

#### **5.10.6.1 Applicability of this Standard to software migration**

If a system or software product (including data) is migrated from an old to a new operational environment, it shall be ensured that any software product or data produced or modified during migration conform to this Standard.

#### **5.10.6.2 Migration planning and execution**

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

- requirements analysis and definition of migration;
- development of migration tools;

- conversion of software product and data;
- migration execution;
- migration verification;
- support for the old environment in the future;
- operator involvement in the activities.

EXPECTED OUTPUT: *Migration plan [MF]*.

#### **5.10.6.3 Contribution to the migration plan**

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

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

EXPECTED OUTPUT: *Migration plan [MF]*.

#### **5.10.6.4 Preparation for migration**

Parallel operations of the old and new environments can be conducted for smooth transition to the new environment. During this period, training shall be provided and specified in the operational plan.

#### **5.10.6.5 Notification of transition to migrated system**

- a. When the scheduled migration takes place, notification shall be sent to all concerned.
- b. All associated old environment's documentation, logs, and code shall be placed in archives.

#### **5.10.6.6 Post-operation review**

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

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

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

#### **5.10.6.7 Maintenance and accessibility of data of former system**

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

### **5.10.7 Software retirement**

#### **5.10.7.1 Retirement planning**

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

- cessation of full or partial support after a certain period of time;
- archiving of the software product and its associated documentation;
- responsibility for any future residual support issues;
- transition to the new software product;
- accessibility of archive copies of data.

EXPECTED OUTPUT: *Retirement plan [MF]*.

**5.10.7.2 Notification of retirement plan**

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

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

EXPECTED OUTPUT: *Retirement notification [MF]*.

**5.10.7.3 Identification of requirements for software retirement**

Parallel operations of the retiring and the new software product can be conducted for smooth transition to the new system. During this period, user training shall be provided as specified in the contract.

**5.10.7.4 Maintenance and accessibility to data of the retired product**

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

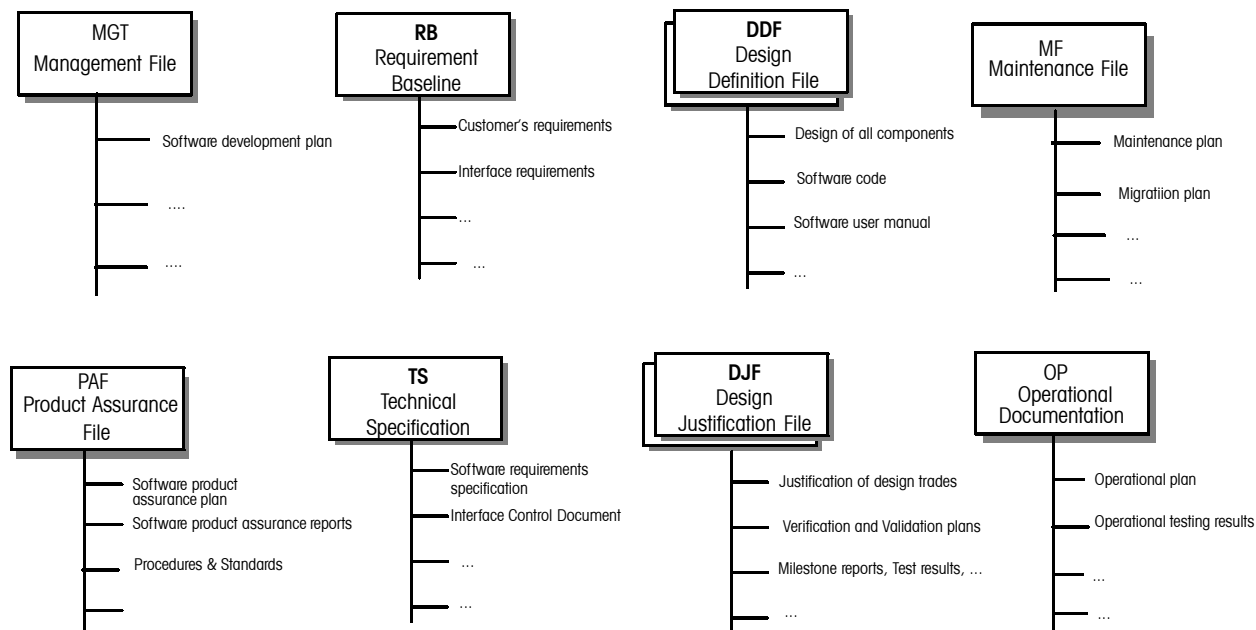
*(This page is intentionally left blank)*

## Annex A (normative)

### Software documentation

#### A.1 Introduction

This annex defines the contents of the software documents to be produced. The contents are defined by the outputs of the clauses in this standard, and the list of the outputs for each milestone of the project is provided below. The detailed structure of the software documents (e.g. table of contents, number of volumes) are not defined here, but left open to be determined by the size and nature of the individual software projects. The overall structure is given in Figure A-1.



**Figure A-1: Overview of software documents**

The following convention is used in this annex to uniquely identify the expected outputs referenced in Clause 5 of this Standard: w.x.y.zindex1-index2 where w.x.y.z is the subclause reference identifier (e.g. 5.4.4.1, 6.2.4.1 or 5.5.1), index1 is used to distinguish an identified separated requirement in a subclause (resulting e.g. from a split of a subclause into several requirements), index2 is used to identify its associated outputs (in case where there are several outputs for a w.x.y.zindex1 subclause).

For example :

- E-40B-5.2.2.1-a denotes the expected output “a” associated to subclause E-40B-5.2.2.1,
- E-40B-5.2.3.2c denotes the expected output associated to subclause 5.2.3.2c, and
- E-40B-5.2.3.9b-a denotes the expected output “a” associated to subclause 5.3.2.9b.

## A.2 Requirements baseline (RB)

### A.2.1 General

The RB expresses the customer’s requirements. It is generated by the requirements engineering processes, and it is the primary input to the SRR review process.

The IRD expresses the customer’s interface requirements for the software to be produced by the supplier. This document is part of the requirements baseline. Depending on the size and nature of the project, the IRD can form separate clauses or separate volumes of the RB.

The IRD shall be produced in all cases where the software product is intended for integration with the customer’s hardware or software products.

### A.2.2 RB contents at SRR

E-40B-5.2.2.1-a	Functions and performance requirements of the system [RB; SRR]
E-40B-5.2.2.1-c	Design constraints [RB; SRR]
E-40B-5.2.2.1-d	Identification of lower level software engineering standards [RB; SRR]
E-40B-5.2.2.1-e	Verification and validation product requirements [RB; SRR]
E-40B-5.2.2.2	Overall safety and reliability requirements of the software to be produced [RB; SRR]
E-40B-5.2.2.3	MMI software mock-up requirements [RB; SRR]
E-40B-5.2.2.4	MMI general requirements [RB; SRR]
E-40B-5.2.3.2c	System partition with definition of items [RB; SRR]
E-40B-5.2.3.2d	System configuration items list [RB; SRR]
E-40B-5.2.4.2	Verification and validation process requirements [RB; SRR]
E-40B-5.2.4.3-a	Functional requirements for support to system and mission level validation [RB; SRR]
E-40B-5.2.4.3-b	Installation and acceptance requirements for the delivered software product at the operational and maintenance site [RB; SRR]
E-40B-5.2.4.5	SRR milestone report [RB; SRR]
E-40B-5.2.5.1a	Software observability requirements [RB; SRR]

E-40B-5.2.5.1b	System observability requirements [RB; SRR]
E-40B-5.2.5.1c	System observability requirements [RB; SRR]
E-40B-5.2.5.1d	System observability requirements [RB; SRR]
E-40B-5.2.5.4	System database specification (content and use) [RB; SRR]
E-40B-5.2.5.5	Development constraints [RB; SRR]
E-40B-5.2.5.6	Requirements for “design for reuse” [RB; SRR]
E-40B-5.2.6.2	Software operations requirements [RB; SRR]
E-40B-5.2.7.1	Software maintenance requirements [RB; SRR]
E-40B-5.2.7.2	Requirements for in flight modification capabilities [RB;SRR]
E-40B-5.3.2.6	Customer approval of requirements baseline [RB; SRR]
E-40B-5.3.4.2-a	Interface management procedures [RB; SRR]
E-40B-5.3.4.2-b	Part of configuration management requirements [RB; SRR]
E-40B-5.3.5.1	Technical budgets and margin philosophy for the project [RB; SRR]
Q-80B-5.6.1	Software procurement process for COTS, OTS or MOTS [RB; SRR]
Q-80B-6.2.2.1	Critical functions identification and analysis [RB; SRR]

#### **A.2.3 RB contents at PDR**

NOTE Not detailed here.

#### **A.2.4 RB contents at CDR**

NOTE Not detailed here.

#### **A.2.5 RB contents at QR**

NOTE Not detailed here.

#### **A.2.6 RB contents at AR**

NOTE Not detailed here.

#### **A.2.7 RB contents at ORR**

NOTE Not detailed here.

#### **A.2.8 IRD contents at SRR**

E-40B-5.2.2.1-b	Interface requirements [IRD(RB); SRR]
E-40B-5.2.3.2a	Software-hardware interface requirements [IRD(RB); SRR]
E-40B-5.2.5.2	System level interface requirements [IRD(RB); SRR]
E-40B-5.2.5.3	System level data interfaces [IRD(RB); SRR]
E-40B-5.2.5.7	System level integration support products [IRD(RB); SRR]
E-40B-5.2.5.8	System level integration preparation requirements [IRD(RB); SRR]
E-40B-5.3.4.1	Interface requirements document [IRD(RB); SRR]

## A.3 Technical specification (TS)

### A.3.1 General

The TS contains the supplier's response to the requirements baseline, and is the primary input to the PDR review process.

Depending on the size and nature of the project, the following sub-documents can be separate chapters or separate volumes of the TS.

The ICD is the supplier's response to the IRD, and is part of the TS.

### A.3.2 TS contents at PDR

- E-40B-5.3.2.7-a Software requirements specification [TS; PDR]
- E-40B-5.3.2.8 Customer approval of technical specification and software architecture [TS, DDF, ICD(TS), DJF; PDR]
- E-40B-5.4.2.1-a Software requirements specification [TS; PDR] - Functional and performance specifications, including hardware characteristics, and environmental conditions under which the software item executes, including budgets requirements [TS; PDR]
- E-40B-5.4.2.1-b Software requirements specification [TS; PDR] - Software product quality requirements (see ECSS-Q-80B subclause 7.2) [TS; PDR]
- E-40B-5.4.2.1-c Software requirements specification [TS; PDR] - Security specifications, including those related to factors which can compromise sensitive information [TS; PDR]
- E-40B-5.4.2.1-d Software requirements specification [TS; PDR] - Human-factors engineering (ergonomics) specifications, including those related to manual operations, human-equipment interactions, constraints on personnel, and areas requiring concentrated human attention, that are sensitive to human errors and training [TS; PDR]
- E-40B-5.4.2.1-e Software requirements specification [TS; PDR] - Data definition and database requirements [TS; PDR]
- E-40B-5.4.2.2-a Specifications for in flight software modifications [TS; PDR]
- E-40B-5.4.2.3 Requirements unique identifier [TS; PDR]
- E-40B-5.4.2.4 Software logical model [TS; PDR]
- E-40B-5.4.2.5 Behavioural view in software logical model for space reactive software [TS; PDR]
- E-40B-5.4.2.6a MMI specifications for software [TS; PDR]
- E-40B-5.4.3.11 Specification of reuse of predeveloped software [TS; PDR]
- Q-80B-6.3.1.4 Software requirements specification - Non-functional requirements [TS; PDR]
- Q-80B-7.1.2,  
Q-80B-7.1.3.3,  
Q-80B-7.2.1.1 Software quality requirements [TS; PDR]



### **A.3.3 TS contents at DDR**

E-40B-5.3.2.9-d Customer approval of the updated technical specifications [TS; DDR]

### **A.3.4 ICD contents at PDR**

E-40B-5.3.2.7-c Interface Control Document [ICD(TS); PDR]

E-40B-5.3.2.8 Customer approval of technical specification and software architecture [TS, DDF, ICD(TS), DJF; PDR]

E-40B-5.4.2.1-f Interfaces external to the software item [ICD(TS); PDR]

E-40B-5.4.3.8-a Preliminary external interfaces design [ICD(TS); PDR]

### **A.3.5 ICD contents at DDR**

E-40B-5.5.2.2-a External interfaces design (update) [ICD(TS); DDR CDR]

### **A.3.6 ICD contents at CDR**

E-40B-5.5.2.2-a External interfaces design (update) [ICD(TS); DDR, CDR]

## **A.4 Design definition file (DDF)**

### **A.4.1 General**

The DDF is a supplier-generated file that documents the result of the design engineering processes. The DDF is the primary input to the CDR review process and it contains all the documents called for by the design engineering requirements.

### **A.4.2 DDF contents at SRR**

NOTE Not detailed here.

### **A.4.3 DDF contents at PDR**

E-40B-5.3.2.7-b Software architectural design [DDF; PDR]

E-40B-5.3.2.8 Customer approval of technical specification and software architecture [TS, DDF, ICD(TS), DJF; PDR]

E-40B-5.4.2.2-b Design for in flight modification [DDF; PDR]

E-40B-5.4.3.1 Software architectural design [DDF; PDR]

E-40B-5.4.3.2 Hierarchy, dependency and interfaces of software components in the software architectural design [DDF; PDR]

E-40B-5.4.3.3 Process, data and control aspects of software components in the software architectural design [DDF; PDR]

E-40B-5.4.3.4-a Software static architecture [DDF; PDR]

E-40B-5.4.3.4-b Software dynamic architecture [DDF; PDR]

E-40B-5.4.3.4-c Software behaviour [DDF; PDR]

E-40B-5.4.3.6a Computational model [DDF; PDR]

E-40B-5.4.3.7 Software dynamic behaviour [DDF; PDR]

E-40B-5.4.3.8-b Preliminary internal interfaces design [DDF; PDR]

Q-80B-6.2.3.5-a,

Q-80B-6.2.3.9-a Measures in the design [DDF; PDR, CDR]

Q-80B-6.2.4.4,

- Q-80B-6.2.4.5 Software configuration file [DDF; PDR, CDR, QR, AR, ORR]
- Q-80B-6.2.4.6 Authorized changes – Software configuration file [DDF; PDR, CDR, QR, AR, ORR]

#### **A.4.4 DDF contents at DDR**

- E-40B-5.3.2.9-a Customer approval of the design definition file (software architectural design, detailed design) [DDF; DDR]
- E-40B-5.5.2.1a Software components design documents [DDF; DDR, CDR]
- E-40B-5.5.2.1b Software components design documents [DDF; DDR, CDR]
- E-40B-5.5.2.1c Software components design documents [DDF; DDR, CDR]
- E-40B-5.5.2.2-b Internal interfaces design (update) [DDF; DDR, CDR]
- E-40B-5.5.2.3-a Software static design [DDF; DDR, CDR]
- E-40B-5.5.2.3-b Software dynamic design [DDF; DDR, CDR]
- E-40B-5.5.2.3-c Software elements behaviour [DDF; DDR, CDR]
- E-40B-5.5.2.5a Software dynamic design [DDF; DDR, CDR]
- E-40B-5.5.2.6 Dynamic behaviour [DDF; DDR, CDR]
- E-40B-5.5.2.7 Compatibility of design methods with the computational model [DDF; DDR, CDR]
- E-40B-5.5.2.8 Software user manual [DDF; DDR, CDR]

#### **A.4.5 DDF contents at CDR**

- E-40B-5.3.2.10b-a Customer approval of the design definition file (e.g. software architectural design, detailed design, code and software user manual) [DDF; CDR]
- E-40B-5.5.2.1a Software components design documents [DDF; DDR, CDR]
- E-40B-5.5.2.1b Software components design documents [DDF; DDR, CDR]
- E-40B-5.5.2.1c Software components design documents [DDF; DDR, CDR]
- E-40B-5.5.2.2-b Internal interfaces design (update) [DDF; DDR, CDR]
- E-40B-5.5.2.3-a Software static design [DDF; DDR, CDR]
- E-40B-5.5.2.3-b Software dynamic design [DDF; DDR, CDR]
- E-40B-5.5.2.3-c Software elements behaviour [DDF; DDR, CDR]
- E-40B-5.5.2.5a Software dynamic design [DDF; DDR, CDR]
- E-40B-5.5.2.6 Dynamic behaviour [DDF; DDR, CDR]
- E-40B-5.5.2.7 Compatibility of design methods with the computational model [DDF; DDR, CDR]
- E-40B-5.5.2.8 Software user manual [DDF; DDR, CDR]
- E-40B-5.5.3.1-a Software component design documents and code (update) [DDF; CDR]
- E-40B-5.5.3.1-b Software configuration file – build procedures [DDF; CDR]
- E-40B-5.5.3.2-a Software component design document and code (update) [DDF; CDR]
- E-40B-5.5.3.3 Software user manual (update) [DDF; CDR]
- E-40B-5.5.4.3 Software user manual (update) [DDF; CDR]
- E-40B-5.6.3.3 Software user manual (update) [DDF; CDR]
- Q-80B-6.2.3.5-a,

- Q-80B-6.2.3.9-a Measures in the design [DDF; PDR, CDR]
- Q-80B-6.2.4.4,
- Q-80B-6.2.4.5 Software configuration file [DDF; PDR, CDR, QR, AR, ORR]
- Q-80B-6.2.4.6 Authorized changes – Software configuration file [DDF; PDR, CDR, QR, AR, ORR]
- Q-80B-6.2.4.8-b Identification and protection method or tool in the software configuration file [DDF; CDR, QR, AR, ORR]
- Q-80B-6.2.4.10 Checksum value in the software configuration file [DDF; CDR, QR, AR, ORR]
- Q-80B-6.2.4.12-b Labels [DDF; CDR, QR, AR]

#### **A.4.6 DDF contents at QR**

- E-40B-5.6.4.3 Software user manual (update) [DDF; QR, AR]
- E-40B-5.6.4.5-a Software release document [DDF; QR]
- E-40B-5.6.4.5-b Software delivery [DDF; QR]
- E-40B-5.7.2.1-a Software delivery [DDF; QR, AR]
- E-40B-5.7.2.1-b Software release document [DDF; QR, AR]
- E-40B-5.7.2.2 Training material [DDF; QR]
- Q-80B-6.2.4.4,
- Q-80B-6.2.4.5 Software configuration file [DDF; PDR, CDR, QR, AR, ORR]
- Q-80B-6.2.4.6 Authorized changes – Software configuration file [DDF; PDR, CDR, QR, AR, ORR]
- Q-80B-6.2.4.8-b Identification and protection method or tool in the software configuration file [DDF; CDR, QR, AR, ORR]
- Q-80B-6.2.4.10 Checksum value in the software configuration file [DDF; CDR, QR, AR, ORR]
- Q-80B-6.2.4.12-b Labels [DDF; CDR, QR, AR]

#### **A.4.7 DDF contents at AR**

- E-40B-5.6.4.3 Software user manual (update) [DDF; QR, AR]
- E-40B-5.7.2.1-a Software delivery [DDF; QR, AR]
- E-40B-5.7.2.1-b Software release document [DDF; QR, AR]
- E-40B-5.7.3.6b-c Software release document [DDF; AR]
- E-40B-5.7.3.6b-d Software delivery [DDF; AR]
- Q-80B-6.2.4.3 Software configuration file [DDF; AR]
- Q-80B-6.2.4.4,
- Q-80B-6.2.4.5 Software configuration file [DDF; PDR, CDR, QR, AR, ORR]
- Q-80B-6.2.4.6 Authorized changes – Software configuration file [DDF; PDR, CDR, QR, AR, ORR]
- Q-80B-6.2.4.8-b Identification and protection method or tool in the software configuration file [DDF; CDR, QR, AR, ORR]
- Q-80B-6.2.4.10 Checksum value in the software configuration file [DDF; CDR, QR, AR, ORR]
- Q-80B-6.2.4.12-b Labels [DDF; CDR, QR, AR]

#### **A.4.8 DDF contents at ORR**

- Q-80B-6.2.4.4,
- Q-80B-6.2.4.5      Software configuration file [DDF; PDR, CDR, QR, AR, ORR]
- Q-80B-6.2.4.6      Authorized changes – Software configuration file [DDF; PDR, CDR, QR, AR, ORR]
- Q-80B-6.2.4.8-b    Identification and protection method or tool in the software configuration file [DDF; PDR, CDR, QR, AR, ORR]
- Q-80B-6.2.4.10     Checksum value in the software configuration file [DDF; CDR, QR, AR, ORR]

### **A.5 Design justification file (DJF)**

#### **A.5.1 General**

The DJF is generated and reviewed at all stages of the development and review processes. It contains the documents that describe the trade-offs, design choice justifications, verification plan, validation plan, validation testing specification, test procedures, test results, evaluations and any other documentation called for to justify the design of the supplier's product. The DJF is a primary input to the CDR, QR and AR milestones, and supporting input for the other milestones.

#### **A.5.2 DJF contents at ANY milestone**

- Q-80B-5.4.5.1-b    Nonconformance [DJF]
- Q-80B-5.4.5.1-c    Nonconformance record [DJF]
- Q-80B-6.2.1.1,
- Q-80B-6.2.1.2,
- Q-80B-6.2.1.3      Software project plans [MGT, MF, DJF]

#### **A.5.3 DJF contents at SRR**

- E-40B-5.2.3.2b    Traceability to system partitioning [DJF; SRR]
- E-40B-5.3.3.2      Milestone review reports [DJF; SRR, PDR, DDR, CDR, QR, AR]
- E-40B-5.3.3.3a    Technical review reports [ DJF; SRR, PDR, DDR, CDR, QR, AR]
- E-40B-5.3.3.3b    Technical review reports [ DJF; SRR, PDR, DDR, CDR, QR, AR]
- Q-80B-5.7.3        Software component list [DJF; SRR,PDR]
- Q-80B-5.8.3.3      Evidence of suitability of the software development environment [DJF; SRR, PDR]
- Q-80B-6.2.6.2-b,
- Q-80B-6.3.6.3,
- Q-80B-6.2.6.4,
- Q-80B-6.2.6.7      Software problem reports [DJF; SRR, PDR, CDR, QR, AR, ORR]
- Q-80B-6.2.6.13    ISVV plan [DJF; SRR, PDR]
- Q-80B-6.2.7.1
- Q-80B-6.2.7.2,
- Q-80B-6.2.7.3,

Q-80B-6.2.7.4	Justification of selection of reused software in the software reuse file [DJF; SRR, PDR].
Q-80B-6.2.7.5, Q-80B-6.2.7.6, Q-80B-6.2.7.7	Software reuse file [DJF; SRR, PDR, CDR]
Q-80B-6.2.7.9	Evaluation report of proposed product in the software reuse file [DJF; SRR, PDR, CDR]
Q-80B-6.2.7.10	Action plan in the software reuse file [DJF; SRR, PDR, CDR]
Q-80B-6.3.4.29-a	ISVV plan [DJF; SRR, PDR]
Q-80B-7.2.3.4	Technical specification for reusable components in the software reuse file [DJF; SRR, PDR]
Q-80B-7.2.3.7	Test reports for the software reuse file [DJF; SRR, PDR]
Q-80B-7.4.1-a, Q-80B-7.4.2, Q-80B-7.4.3, Q-80B-7.4.5	Justification of selection of ground equipment [DJF; SRR, PDR]
Q-80B-7.4.1-b	Receiving inspection report [DJF; SRR, PDR]

#### **A.5.4 DJF contents at PDR**

E-40B-5.3.2.7-d	Software architectural design trade-offs [DJF; PDR]
E-40B-5.3.2.8	Customer approval of technical specification and software architecture [TS, DDF, ICD(TS), DJF; PDR]
E-40B-5.3.3.2	Milestone review reports [DJF; SRR, PDR, DDR, CDR, QR, AR]
E-40B-5.3.3.3a	Technical review reports [DJF; SRR, PDR, DDR, CDR, QR, AR]
E-40B-5.3.3.3b	Technical review reports [DJF; SRR, PDR, DDR, CDR, QR, AR]
E-40B-5.3.5.2	Margins and technical budgets status [DJF; PDR, DDR, CDR, QR, AR]
E-40B-5.4.2.6b	Report on evaluation of MMI specifications using a software mock-up [DJF; PDR]
E-40B-5.4.3.6b	Scheduling simulation report [DJF; PDR]
E-40B-5.4.3.9	Software for intended reuse – justification of methods and tools [DJF; PDR]
E-40B-5.4.3.10	Software for intended reuse – evaluation of reuse potential [DJF; PDR, CDR]
E-40B-5.4.3.12	Justification of reuse with respect to requirements baseline in the software reuse file [DJF; PDR]
E-40B-5.4.3.13	Software integration test plan (preliminary) [DJF; PDR]
E-40B-5.4.3.14	PDR milestone report [DJF; PDR]
E-40B-5.8.2.1a	Software verification plan – organizational independence and effort identification [DJF; PDR]
E-40B-5.8.2.1b	Software verification plan – identification of risks and level of independence [DJF; PDR]

E-40B-5.8.2.1c	Software verification plan – criticality and resources identification [DJF; PDR]
E-40B-5.8.2.2a	Software verification plan – verification process identification [DJF; PDR]
E-40B-5.8.2.2b	Software verification plan – software products identification [DJF; PDR]
E-40B-5.8.2.2c	Software verification plan – methods and tools [DJF; PDR]
E-40B-5.8.2.3a	Independent verification organization selection [DJF; PDR]
E-40B-5.8.2.3b	Appropriate elements of project requirements dealing with project organization [DJF; PDR]
E-40B-5.8.2.4	Software verification plan – organization and activities [DJF; PDR]
E-40B-5.8.3.1-a	Requirements traceability matrices [DJF; PDR]
E-40B-5.8.3.1-b	Requirements verification report [DJF; PDR]
E-40B-5.8.3.2-a	Software architectural design to requirements traceability matrices [DJF; PDR]
E-40B-5.8.3.2-b	Software architectural design and interface verification report [DJF; PDR]
E-40B-5.8.3.6	Software documentation verification report [DJF; PDR, CDR, QR]
E-40B-5.8.3.11a	Schedulability analysis or design feasibility verification with models or simulation [DJF; PDR]
E-40B-5.8.3.12a	Technical budgets – memory and CPU estimation [DJF; PDR]
E-40B-5.8.3.13a	Software behaviour verification [DJF; PDR]
E-40B-5.6.2.1	Software validation plan – effort and independence [DJF; PDR]
E-40B-5.6.2.2a	Software validation plan – validation process identification
E-40B-5.6.2.2b	Software validation plan – methods and tools [DJF; PDR]
E-40B-5.6.2.3a	Independent software validation plan – organization selection [DJF; PDR]
E-40B-5.6.2.3b	Independent software validation plan – level of independence [DJF; PDR]
E-40B-5.6.2.4	Software validation plan – organization and activities [DJF; PDR]
Q-80B-5.7.3	Software component list [DJF; SRR, PDR]
Q-80B-5.8.3.3	Evidence of suitability of the software development environment [DJF; SRR, PDR]
Q-80B-6.2.3.2	Verification and validation documentation [DJF; PDR, CDR, QR, AR]
Q-80B-6.2.3.5-b	Verification and validation documentation [DJF; PDR, CDR]
Q-80B-6.2.3.9-b	Verification and validation documentation [DJF; PDR, CDR].
Q-80B-6.2.6.1	Software verification plan – verification of quality requirements [DJF; PDR].
Q-80B-6.2.6.2-b,	

Q-80B-6.2.6.3,	
Q-80B-6.2.6.4,	
Q-80B-6.2.6.7	Software problem reports [DJF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-6.2.6.13	ISVV plan [DJF; SRR, PDR] and ISVV report [DJF; PDR, CDR, QR, AR, ORR]
Q-80B-6.2.7.1,	
Q-80B-6.2.7.2,	
Q-80B-6.2.7.3,	
Q-80B-6.2.7.4	Justification of selection of reused software in the software reuse file [DJF; SRR, PDR]
Q-80B-6.2.7.5,	
Q-80B-6.2.7.6,	
Q-80B-6.2.7.7	Software reuse file [DJF; SRR, PDR, CDR]
Q-80B-6.2.7.9	Evaluation report of proposed product in the software reuse file [DJF; SRR, PDR, CDR]
Q-80B-6.2.7.10	Action plan in the software reuse file [DJF; SRR, PDR, CDR]
Q-80B-6.3.3.5	Document justifying suitability of language [DJF; PDR]
Q-80B-6.3.4.21,	
Q-80B-6.3.4.23,	
Q-80B-6.3.4.24,	
Q-80B-6.3.4.25,	
Q-80B-6.3.4.26	Contribution to the test and validation documentation [DJF; PDR, CDR, QR, AR, ORR].
Q-80B-6.3.4.29-a	ISVV plan [DJF; SRR, PDR]
Q-80B-6.3.4.29-b	ISVV report [DJF; PDR, CDR, QR, AR, ORR]
Q-80B-7.1.10	Numerical accuracy analysis [DJF; PDR, CDR, QR]
Q-80B-7.2.1.4	Verification and validation method for each requirement [DJF; PDR]
Q-80B-7.2.3.3-b	Justification of design choices [DJF; PDR, CDR].
Q-80B-7.2.3.4	Technical specification for reusable components in the software reuse file [DJF; SRR, PDR]
Q-80B-7.2.3.7	Test reports for the software reuse file [DJF; SRR, PDR]
Q-80B-7.4.1-a,	
Q-80B-7.4.2,	
Q-80B-7.4.3,	
Q-80B-7.4.5	Justification of selection of ground equipment [DJF; SRR, PDR]
Q-80B-7.4.1-b	Receiving inspection report [DJF; SRR, PDR]

#### **A.5.5 DJF contents of DDR**

E-40B-5.3.2.9-b	Customer approval of the design of software interface and the software integration test plan [DJF; DDR]
E-40B-5.3.2.9-c	Customer approval of the margins and technical budget status [DJF; DDR]



E-40B-5.3.3.2	Milestone review reports [DJF; SRR, PDR, DDR, CDR, QR, AR]
E-40B-5.3.3.3a	Technical review reports [DJF; SRR, PDR, DDR, CDR, QR, AR]
E-40B-5.3.3.3b	Technical review reports [DJF; SRR, PDR, DDR, CDR, QR, AR]
E-40B-5.3.5.2	Margins and technical budgets status [DJF; PDR, DDR, CDR, QR, AR]
E-40B-5.5.2.5b	Scheduling simulation report [DJF; DDR, CDR]
E-40B-5.5.2.9	Software unit test plan [DJF; DDR, CDR]
E-40B-5.5.2.10	Software integration test plan (update) [DJF; DDR, CDR]
E-40B-5.5.2.11	DDR milestone report [DJF; DDR]
E-40B-5.8.3.11b	Schedulability analysis (update) [DJF; DDR, CDR]
E-40B-5.8.3.12b	Technical budgets – memory and CPU estimation [DJF; DDR, CDR]
E-40B-5.8.3.13b	Software behaviour verification [DJF; DDR, CDR]

#### **A.5.6 DJF contents at CDR**

E-40B-5.3.2.10a	CDR milestone report [DJF; CDR]
E-40B-5.3.2.10b-b	Customer approval of the design justification file (e.g. software unit test plan, software integration test plan, results of unit and integration tests and results of validation with respect to the technical specifications) [DJF; CDR]
E-40B-5.3.3.2	Milestone review reports [DJF; SRR, PDR, DDR, CDR, QR, AR]
E-40B-5.3.3.3a	Technical review reports [DJF; SRR, PDR, DDR, CDR, QR, AR]
E-40B-5.3.3.3b	Technical review reports [DJF; SRR, PDR, DDR, CDR, QR, AR]
E-40B-5.3.5.2	Margins and technical budgets status [DJF; PDR, DDR, CDR, QR, AR]
E-40B-5.4.3.10	Software for intended reuse – evaluation of reuse potential [DJF; PDR, CDR]
E-40B-5.5.2.5b	Scheduling simulation report [DJF; CDR]
E-40B-5.5.2.9	Software unit test plan [DJF; DDR, CDR]
E-40B-5.5.2.10	Software integration test plan (update) [DJF; DDR, CDR]
E-40B-5.5.3.1-c	Software unit test plan (update) [DJF; CDR]
E-40B-5.5.3.2-b	Software unit test reports [DJF; CDR]
E-40B-5.5.3.4	Software integration test plan (update) [DJF; CDR]
E-40B-5.5.4.1	Software integration test plan [DJF; CDR]
E-40B-5.5.4.2	Software integration test report [DJF; CDR]
E-40B-5.8.3.3-a	Detailed design traceability matrices [DJF; CDR]
E-40B-5.8.3.3-b	Detailed design verification report [DJF; CDR]
E-40B-5.8.3.4-a	Software code traceability matrices [DJF; CDR]
E-40B-5.8.3.4-b	Software code verification report [DJF; CDR]
E-40B-5.8.3.5a	Software integration verification report [DJF; CDR]



E-40B-5.8.3.5b	Software integration verification report [DJF; CDR]
E-40B-5.8.3.6	Software documentation verification report [DJF; PDR, CDR, QR]
E-40B-5.8.3.7-b	Traceability of the technical specification to the validation tests [DJF; CDR]
E-40B-5.8.3.8-a	Validation report evaluation with respect to the technical specification [DJF; CDR]
E-40B-5.8.3.10a	Problem and nonconformance reports [DJF; CDR, QR, AR]
E-40B-5.8.3.10b	Problem and nonconformance reports [DJF; CDR, QR, AR]
E-40B-5.8.3.11b	Schedulability analysis (update) [DJF; DDR, CDR]
E-40B-5.8.3.11c	Schedulability analysis (update) [DJF; CDR]
E-40B-5.8.3.12b	Technical budgets - memory and CPU estimation [DJF; DDR, CDR]
E-40B-5.8.3.12c	Technical budgets (update) - memory and CPU calculation [DJF; CDR]
E-40B-5.8.3.13b	Software behaviour verification [DJF; DDR, CDR]
E-40B-5.8.3.14	Testing feasibility report [DJF; CDR]
E-40B-5.6.3.1-a	Software validation with respect to the technical specification testing specification [DJF; CDR]
E-40B-5.6.3.1-b	Verification report [DJF; CDR]
E-40B-5.6.3.2	Validation testing report with respect to the technical specification [DJF; CDR]
E-40B-5.6.3.5	CDR milestone report [DJF; CDR]
Q-80B-6.2.3.2	Verification and validation documentation [DJF; PDR, CDR, QR, AR]
Q-80B-6.2.3.5-b	Verification and validation documentation [DJF; PDR, CDR]
Q-80B-6.2.3.9-b	Verification and validation documentation [DJF; PDR, CDR]
Q-80B-6.2.6.2-b, Q-80B-6.2.6.3,	
Q-80B-6.2.6.4	Software problem reports [DJF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-6.2.6.5-b, Q-80B-6.2.6.6-b	Software problem reports [DJF; CDR, QR, AR, ORR]
Q-80B-6.2.6.7	Software problem reports [DJF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-6.2.6.13	ISVV report [DJF; PDR, CDR, QR, AR, ORR]
Q-80B-6.2.7.5, Q-80B-6.2.7.6,	
Q-80B-6.2.7.7	Software reuse file [DJF; SRR, PDR, CDR]
Q-80B-6.2.7.9	Evaluation report of proposed product in the software reuse file [DJF; SRR, PDR, CDR]
Q-80B-6.2.7.10	Action plan in the software reuse file [DJF; SRR, PDR, CDR]

Q-80B-6.3.3.10	Justification for retention of unreachable code [DJF; CDR, QR, AR]
Q-80B-6.3.4.7	Nonconformance report and SPR [DJF; CDR, QR, AR, ORR]
Q-80B-6.3.4.17, Q-80B-6.3.4.18, Q-80B-6.3.4.19	Updated test documentation [DJF; CDR, QR, AR, ORR]
Q-80B-6.3.4.21, Q-80B-6.3.4.23, Q-80B-6.3.4.24, Q-80B-6.3.4.25, Q-80B-6.3.4.26	Contribution to the test and validation documentation [DJF; PDR, CDR, QR, AR, ORR]
Q-80B-6.3.4.29-b	ISVV report [DJF; PDR, CDR, QR, AR, ORR]
Q-80B-6.3.4.30-b, Q-80B-6.3.4.31-b, Q-80B-6.3.4.32-b	Software problem reports [DJF; CDR, QR, AR, ORR]
Q-80B-7.1.10	Numerical accuracy analysis [DJF; PDR, CDR, QR]
Q-80B-7.2.2.3-b	Justification of design choices [DJF; PDR, CDR]
Q-80B-7.2.3.6	Test reports for the software reuse file [DJF; CDR]
Q-80B-7.3.1.6	Verification reports [DJF; CDR, QR, AR]

#### **A.5.7 DJF contents at QR**

E-40B-5.3.2.12a	QR milestone report [DJF; QR]
E-40B-5.3.2.12b	Customer's approval of qualified state [DJF; QR]
E-40B-5.3.3.2	Milestone review reports [DJF; SRR, PDR, DDR, CDR, QR, AR]
E-40B-5.3.3.3a	Technical review reports [DJF; SRR, PDR, DDR, CDR, QR, AR]
E-40B-5.3.3.3b	Technical review reports [DJF; SRR, PDR, DDR, CDR, QR, AR]
E-40B-5.3.5.2	Margins and technical budgets status [DJF; PDR, DDR, CDR, QR, AR]
E-40B-5.8.3.7-a	Traceability of the requirements baseline to the validation tests [DJF; QR, AR]
E-40B-5.8.3.8-b	Validation report evaluation with respect to the requirements baseline [DJF; QR, AR]
E-40B-5.8.3.10a	Problem and nonconformance reports [DJF; CDR, QR, AR]
E-40B-5.8.3.10b	Problem and nonconformance reports [DJF; CDR, QR, AR]
E-40B-5.6.4.1-a	Software validation with respect to the requirements baseline testing specification [DJF; QR, AR]
E-40B-5.6.4.1-b	Verification report [DJF; QR, AR]
E-40B-5.6.4.2	Validation testing report with respect to the requirements baseline [DJF; QR, AR]
E-40B-5.6.4.5.-c	Software design and test evaluation report [DJF; QR]
E-40B-5.6.4.5.-d	Validation testing report [DJF; QR]

E-40B-5.6.4.5.-e	Test specification evaluation [DJF; QR]
E-40B-5.6.4.5.-f	QR milestone report [DJF; QR]
Q-80B-6.2.3.2	Verification and validation documentation [DJF; PDR, CDR, QR, AR]
Q-80B-6.2.6.2-b,	
Q-80B-6.2.6.3,	
Q-80B-6.2.6.4	Software problem reports [DJF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-6.2.6.5-b,	
Q-80B-6.2.6.6-b	Software problem reports [DJF; CDR, QR, AR, ORR]
Q-80B-6.2.6.7	Software problem reports [DJF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-6.2.6.13	ISVV report [DJF; PDR, CDR, QR, AR, ORR]
Q-80B-6.3.3.10	Justification for retention of unreachable code [DJF; CDR, QR, AR]
Q-80B-6.3.4.7	Nonconformance report and SPR [DJF; CDR, QR, AR, ORR]
Q-80B-6.3.4.17,	
Q-80B-6.3.4.18,	
Q-80B-6.3.4.19	Updated test documentation [DJF; CDR, QR, AR, ORR]
Q-80B-6.3.4.21,	
Q-80B-6.3.4.23,	
Q-80B-6.3.4.24,	
Q-80B-6.3.4.25,	
Q-80B-6.3.4.26	Contribution to the test and validation documentation [DJF; PDR, CDR, QR, AR, ORR]
Q-80B-6.3.4.29-b	ISVV report [DJF; PDR, CDR, QR, AR, ORR]
Q-80B-6.3.4.30-b,	
Q-80B-6.3.4.31-b,	
Q-80B-6.3.4.32-b	Software problem reports [DJF; CDR, QR, AR, ORR]
Q-80B-7.1.10	Numerical accuracy analysis [DJF; PDR, CDR, QR]
Q-80B-7.3.1.6	Verification reports [DJF; CDR, QR, AR]

#### **A.5.8 DJF contents at AR**

E-40B-5.3.2.14	Customer's approval of accepted state [DJF; AR].
E-40B-5.3.3.2	Milestone review reports [DJF; SRR, PDR, DDR, CDR, QR, AR]
E-40B-5.3.3.3a	Technical review reports [DJF; SRR, PDR, DDR, CDR, QR, AR]
E-40B-5.3.3.3b	Technical review reports [DJF; SRR, PDR, DDR, CDR, QR, AR]
E-40B-5.3.5.2	Margins and technical budgets status [DJF; PDR, DDR, CDR, QR, AR]
E-40B-5.7.2.3	Installation plan [DJF; AR]
E-40B-5.7.2.4	Installation report [DJF; AR]

E-40B-5.7.3.1	Acceptance test plan [DJF; AR]
E-40B-5.7.3.2	Acceptance test report [DJF; AR]
E-40B-5.7.3.3	Executable code generation test in the acceptance test plan [DJF; AR]
E-40B-5.7.3.4a	AR milestone report [DJF; AR]
E-40B-5.7.3.4b	AR milestone report [DJF; AR]
E-40B-5.7.3.4c	Acceptance testing documentation [DJF; AR]
E-40B-5.7.3.5	Traceability of acceptance tests to the requirements baseline [DJF; AR]
E-40B-5.7.3.6a	AR milestone report [DJF; AR]
E-40B-5.7.3.6b-a	Software acceptance data package [DJF; AR]
E-40B-5.7.3.6b-b	Acceptance testing documentation [DJF; AR]
E-40B-5.8.3.7-a	Traceability of the requirements baseline to the validation tests [DJF; QR, AR]
E-40B-5.8.3.8-b	Validation report evaluation with respect to the requirements baseline [DJF; QR, AR]
E-40B-5.8.3.9	Complement of validation at system level [DJF; AR]
E-40B-5.8.3.10a	Problem and nonconformance reports [DJF; CDR, QR, AR]
E-40B-5.8.3.10b	Problem and nonconformance reports [DJF; CDR, QR, AR]
E-40B-5.6.4.1-a	Software validation with respect to the requirements baseline testing specification [DJF; QR, AR]
E-40B-5.6.4.1-b	Verification report [DJF; QR, AR]
E-40B-5.6.4.2	Validation testing report with respect to the requirements baseline [DJF; QR, AR]
Q-80B-6.2.3.2	Verification and validation documentation [DJF; PDR, CDR, QR, AR]
Q-80B-6.2.6.2-b, Q-80B-6.2.6.3, Q-80B-6.2.6.4	Software problem reports [DJF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-6.2.6.5-b, Q-80B-6.2.6.6-b	Software problem reports [DJF; CDR, QR, AR, ORR]
Q-80B-6.2.6.7	Software problem reports [DJF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-6.2.6.13	ISVV report [DJF; PDR, CDR, QR, AR, ORR]
Q-80B-6.3.3.10	Justification for retention of unreachable code [DJF; CDR, QR, AR]
Q-80B-6.3.4.7	Nonconformance report and SPR [DJF; CDR, QR, AR, ORR]
Q-80B-6.3.4.17, Q-80B-6.3.4.18, Q-80B-6.3.4.19 Q-80B-6.3.4.21, Q-80B-6.3.4.23, Q-80B-6.3.4.24,	Updated test documentation [DJF; CDR, QR, AR, ORR]

- Q-80B-6.3.4.25,
- Q-80B-6.3.4.26 Contribution to the test and validation documentation [DJF; PDR, CDR, QR, AR, ORR]
- Q-80B-6.3.4.28 Contribution to the test and validation documentation [DJF; AR]
- Q-80B-6.3.4.29-b ISVV report [DJF; PDR, CDR, QR, AR, ORR]
- Q-80B-6.3.4.30-b,
- Q-80B-6.3.4.31-b,
- Q-80B-6.3.4.32-b Software problem reports [DJF; CDR, QR, AR, ORR]
- Q-80B-6.3.5.1 Contribution to the installation plan [DJF; AR]
- Q-80B-6.3.5.3,
- Q-80B-6.3.5.5 Acceptance test plan [DJF; AR]
- Q-80B-6.3.5.7 Nonconformance report [DJF; AR]
- Q-80B-6.3.5.8,
- Q-80B-6.3.5.9 Acceptance test report [DJF; AR]
- Q-80B-6.3.6.1 Contribution to the operational plan [DJF; AR]
- Q-80B-7.3.1.6 Verification reports [DJF, CDR, QR, AR]

#### **A.5.9 DJF contents at ORR**

- Q-80B-6.2.6.2-b,
- Q-80B-6.2.6.3,
- Q-80B-6.2.6.4 Software problem reports [DJF; SRR, PDR, CDR, QR, AR, ORR]
- Q-80B-6.2.6.5-b,
- Q-80B-6.2.6.6-b Software problem reports [DJF; CDR, QR, AR, ORR]
- Q-80B-6.2.6.7 Software problem reports [DJF; SRR, PDR, CDR, QR, AR, ORR]
- Q-80B-6.2.6.13 ISVV report [DJF; PDR, CDR, QR, AR, ORR]
- Q-80B-6.3.4.7 Nonconformance report and SPR [DJF; CDR, QR, AR, ORR]
- Q-80B-6.3.4.17,
- Q-80B-6.3.4.18,
- Q-80B-6.3.4.19 Updated test documentation [DJF; CDR, QR, AR, ORR]
- Q-80B-6.3.4.21,
- Q-80B-6.3.4.23,
- Q-80B-6.3.4.24,
- Q-80B-6.3.4.25,
- Q-80B-6.3.4.26 Contribution to the test and validation documentation [DJF; PDR, CDR, QR, AR, ORR]
- Q-80B-6.3.4.29-b ISVV report [DJF; PDR, CDR, QR, AR, ORR]
- Q-80B-6.3.4.30-b,
- Q-80B-6.3.4.31-b,
- Q-80B-6.3.4.32-b Software problem reports [DJF; CDR, QR, AR, ORR]

## A.6 Management file (MGT)

### A.6.1 General

The MGT is a supplier generated file that describes the management features of the software project (for instance, organizational breakdown and responsibilities, work activities breakdown, selected life cycle, deliveries, milestones and risks).

### A.6.2 MGT at ANY milestone

Q-80B-6.2.1.1,

Q-80B-6.2.1.2,

Q-80B-6.2.1.3      Software project plans [MGT, MF, DJF]

### A.6.3 MGT contents at SRR

E-40B-5.2.5.9      System level integration support requirements [MGT; SRR]

E-40B-5.3.2.1      Definition of the software life cycle phases included in the software development plan [MGT; SRR, PDR]

E-40B-5.3.2.2a      Project software development life cycle definition, included in the software development plan [MGT; SRR, PDR]

E-40B-5.3.2.2b      Definition of software development, operations and maintenance techniques and identification of project risks, included in the software development plan [MGT; SRR, PDR]

E-40B-5.3.2.2c      Definition of software life cycle in line with the software and system level processes included in the software development plan [MGT; SRR, PDR]

E-40B-5.3.2.3      Review plan - milestones (included in the software development plan) [MGT; SRR, PDR]

E-40B-5.3.2.4      Identification of outputs at each milestone (included in the software development plan) [MGT; SRR, PDR]

E-40B-5.3.2.11      Software verification and validation activities phasing in the software development plan [MGT; SRR, PDR]

E-40B-5.3.2.14      Phasing of activities of the software validation with respect to the requirements baseline in the software development plan [MGT; SRR, PDR]

E-40B-5.3.2.15      Software procurement process implementation in the software development plan [MGT; SRR, PDR]

Q-80B-5.2.5.1      Training plan [MGT; SRR]

Q-80B-5.4.5.2      Identification of SRB and members [MGT; SRR]

Q-80B-5.7.4      Procurement data [MGT; SRR, PDR]

Q-80B-5.8.2      Descriptions of choices of development equipment in the software development plan [MGT; SRR, PDR]

Q-80B-6.2.4.2      Software configuration management [MGT; SRR, PDR]

Q-80B-6.3.2.1      Definition of methodology and tools in the software development plan [MGT; SRR, PDR]

Q-80B-7.2.3.5      Configuration management for software reuse file [MGT; SRR, PDR]

#### **A.6.4 MGT contents at PDR**

E-40B-5.3.2.1	Definition of the software life cycle phases included in the software development plan [MGT; SRR, PDR]
E-40B-5.3.2.2a	Project software development life cycle definition, included in the software project development plan [MGT; SRR, PDR]
E-40B-5.3.2.2b	Definition of software development, operations and maintenance techniques and identification of project risks, included in the software development plan [MGT; SRR, PDR]
E-40B-5.3.2.2c	Definition of software life cycle in line with the software and system level processes included in the software development plan [MGT; SRR, PDR]
E-40B-5.3.2.3	Review plan - milestones (included in the software development plan [MGT; SRR, PDR]
E-40B-5.3.2.4	Identification of outputs at each milestone (included in the software development plan) [MGT; SRR, PDR]
E-40B-5.2.3.5	Elements of the software maintenance in the software development plan [MGT; PDR]
E-40B-5.3.2.11	Software verification and validation activities phasing in the software development plan [MGT; SRR, PDR]
E-40B-5.3.2.14	Phasing of activities of the software validation with respect to the requirements baseline in the software development plan [MGT; SRR, PDR]
E-40B-5.3.2.15	Software procurement process implementation in the software development plan [MGT; SRR, PDR]
Q-80B-5.7.3.2	Procurement data [MGT; SRR, PDR]
Q-80B-5.8.2	Descriptions of choices of development equipment in the software development plan [MGT; SRR, PDR]
Q-80B-6.2.4.2	Software configuration management [MGT; SRR, PDR]
Q-80B-6.3.2.1	Definition of methodology and tools in the software development plan [MGT; SRR, PDR]
Q-80B-7.2.3.5	Configuration management for software reuse file [MGT; SRR, PDR]

#### **A.6.5 MGT contents at CDR**

NOTE Not detailed here.

#### **A.6.6 MGT contents at QR**

NOTE Not detailed here.

#### **A.6.7 MGT contents at AR**

NOTE Not detailed here.

#### **A.6.8 MGT contents at ORR**

NOTE Not detailed here.

## **A.7 Maintenance file (MF)**

### **A.7.1 General**

The MF is a maintainer generated file that describes the planning and status of the maintenance, migration and retirement activities

### **A.7.2 MF contents at ANY milestone**

E-40B-5.10.2.3b	Problem and nonconformance report [MF]
E-40B-5.10.3.1	Modification analysis report and problem analysis report [MF]
E-40B-5.10.3.2	Modification analysis report and problem analysis report [MF]
E-40B-5.10.3.3	Modification analysis report and problem analysis report [MF]
E-40B-5.10.3.4	Modification analysis report and problem analysis report [MF]
E-40B-5.10.4.1	Modification identification [MF]
E-40B-5.10.5a	Baseline for changes [MF]
E-40B-5.10.5b	Baseline for changes [MF]
E-40B-5.10.6.2	Migration plan [MF]
E-40B-5.10..3	Migration plan [MF]
E-40B-5.10.7.1	Retirement plan [MF]
E-40B-5.10.7.2	Retirement notification [MF]
Q-80B-6.2.1.1, Q-80B-6.2.1.2,	
Q-80B-6.2.1.3	Software project plans [MGT, MF, DJF]

### **A.7.3 MF contents at PDR**

NOTE Not detailed here.

### **A.7.4 MF contents at QR**

E-40B-5.10.2.1	Maintenance plan - plans and procedures [MF; QR]
E-40B-5.10.2.2	Maintenance plan - applicability of development process procedures, methods, tools and standards [MF; QR]
E-40B-5.10.2.3a	Maintenance plan - problem reporting and handling [MF; QR]
E-40B-5.10.2.4	Maintenance plan - configuration management process [MF; QR]
E-40B-5.10.2.5a	Maintenance plan [MF; QR]
E-40B-5.10.2.5b	Long term maintenance solutions [MF; QR]
Q-80B-6.3.7.1, Q-80B-6.3.7.2,	
Q-80B-6.3.7.4	Maintenance plan - assurance [MF; QR, AR, ORR]
Q-80B-6.3.7.5	Rules for submission of maintenance reports - maintenance plan - assurance [MF; QR, AR, ORR]
Q-80B-6.3.7.6, Q-80B-6.3.7.7	Maintenance records [MF; QR, AR, ORR]



### **A.7.5 MF contents at AR**

- Q-80B-6.3.7.1,
- Q-80B-6.3.7.2,
- Q-80B-6.3.7.4 Maintenance plan – assurance [MF; QR, AR, ORR]
- Q-80B-6.3.7.5 Rules for submission of maintenance reports – Maintenance plan – assurance [MF; QR, AR, ORR]
- Q-80B-6.3.7.6,
- Q-80B-6.3.7.7 Maintenance records [MF; QR, AR, ORR]

### **A.7.6 MF contents at ORR**

- Q-80B-6.3.7.1,
- Q-80B-6.3.7.2,
- Q-80B-6.3.7.4 Maintenance plan – assurance [MF; QR, AR, ORR]
- Q-80B-6.3.7.5 Rules for submission of maintenance reports – Maintenance plan – assurance [MF; QR, AR, ORR]
- Q-80B-6.3.7.6,
- Q-80B-6.3.7.7 Maintenance records [MF; QR, AR, ORR]

## **A.8 Operational documentation (OP)**

### **A.8.1 General**

The operation process is a system level activity, defined by the customer's requirements for the space system. The corresponding software engineering process is therefore not independent engineering activity, but is a support process at system level. Hence, the outputs of the process are contributions to system level outputs, and the outputs below are therefore either integrated with the software development documentation, or controlled and developed as part of a system documentation tree. The outputs are identified and grouped below. The system level documentation tree defines how the documents are included.

### **A.8.2 OP contents at ANY review**

- E-40B-5.9.2.2b Problem and nonconformance report [OP]
- E-40B-5.9.5.1b Record of the user's request and subsequent actions [OP]
- E-40B-5.10.6.6a Post operation review report [OP]

### **A.8.3 OP contents at ORR**

- E-40B-5.2.6.1 Operational plan [OP; ORR]
- E-40B-5.9.2.1 Operational plan – plan and standards [OP; ORR]
- E-40B-5.9.2.2a Operational plan – procedures for problem handling [OP; ORR]
- E-40B-5.9.2.3 Operational plan – operational testing specifications [OP; ORR]
- E-40B-5.9.3.1a Operational testing results [OP; ORR]
- E-40B-5.9.3.1b Software delivery [OP; ORR]
- E-40B-5.9.3.2a Validation of operational requirements [OP; ORR]
- E-40B-5.9.3.2b Demonstration criteria [OP; ORR]

## A.9 Product assurance file (PAF)

### A.9.1 General

The PAF contains documents related to planning and definition of requirements and activities related to software product assurance activities and processes. Depending on the size and nature of the project, the following sub-documents can be separate clauses or separate volumes of the PAF.

### A.9.2 PAF contents at ANY milestone

Q-80B-5.2.5.2	Record of training and experience [PAF]
Q-80B-5.4.4-a	Preliminary alert information [PAF]
Q-80B-5.4.4-b	Alert information [PAF]
Q-80B-5.6.1-a	Results of pre-award audits [PAF]
Q-80B-5.6.1-b	Records of procurement sources [PAF]
Q-80B-5.9.1	Software process assessment plan [PAF]
Q-80B-5.9.2	Software process assessment procedure [PAF]
Q-80B-5.9.3	Software process assessment records [PAF]
Q-80B-5.9.4	Software process assessment records: strengths and weaknesses [PAF]
Q-80B-5.9.5	Software process assessment records: quality data [PAF]
Q-80B-5.9.6	Software process assessment records: improvement plan [PAF]
Q-80B-5.9.7	Software process assessment records: updates to process or project documentation [PAF]

### A.9.3 PAF contents at SRR

Q-80B-5.2.2-a	Responsibility, authority and interrelation of personnel managing, performing and verifying work affecting quality [PAF; SRR]
Q-80B-5.2.2-b	External and internal interfaces and responsibilities of each organization [PAF; SRR]
Q-80B-5.2.2-c	Lower level supplier performing delegated product assurance tasks [PAF; SRR]
Q-80B-5.2.3	Software product assurance resource requirements [PAF; SRR]
Q-80B-5.2.4.1	Identification of the software product assurance manager [PAF; SRR]
Q-80B-5.4.1.1, Q-80B-5.4.1.2,	
Q-80B-5.4.1.4	Software product assurance plan [PAF; SRR, PDR]
Q-80B-5.4.1.3	Software product assurance plan [PAF; CDR, QR, AR, ORR]
Q-80B-5.4.1.6	Compliance matrix [PAF; SRR, PDR]
Q-80B-5.4.2.2	Assessment of the quality of software development process in the software product assurance report [PAF; SRR, PDR, CDR, QR, AR, ORR]

Q-80B-5.4.2.3	Assessment of the quality of software product in the software product assurance report [PAF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-5.4.2.4	Software product assurance report [PAF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-5.4.3	Audit plan and schedule [PAF; SRR]
Q-80B-5.4.5.1-a	Description of nonconformance control system [PAF; SRR]
Q-80B-5.4.5.3	Identification of the point in the software life cycle from which the nonconformance procedures apply - software product assurance plan [PAF; SRR, PDR]
Q-80B-5.6.2.1, Q-80B-5.6.2.2	Software product assurance requirements for suppliers [PAF; SRR, PDR]
Q-80B-5.6.4	Evidence of dependability and safety criticality classification [PAF; SRR]
Q-80B-5.8.3.5	Software product assurance report [PAF; SRR, PDR, CDR, QR, AR, ORR].
Q-80B-6.1.1, Q-80B-6.1.5, Q-80B-6.1.6, Q-80B-6.1.7, Q-80B-6.1.8, Q-80B-6.1.9	Reference to software development life cycle definition [PAF; SRR, PDR]
Q-80B-6.2.1.4	Identification of software plans, their interrelations and schedule for preparation [PAF; SRR, PDR].
Q-80B-6.2.2.2-a	Software criticality analysis report [PAF; SRR, PDR]
Q-80B-6.2.2.2-b	Input to the system safety data package [PAF; SRR, PDR]
Q-80B-6.2.4.8-a, Q-80B-6.2.4.9, Q-80B-6.2.4.11	Identification and protection method or tool [PAF; SRR, PDR]
Q-80B-6.2.4.12-a	Labelling method [PAF; SRR, PDR].
Q-80B-6.2.5.1, Q-80B-6.2.5.2, Q-80B-6.2.5.3	Details of metrics in software product assurance plan [PAF; SRR, PDR]
Q-80B-6.2.6.2-a Q-80B-6.2.6.8, Q-80B-6.2.6.9	SPA reports [PAF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-6.2.6.11, Q-80B-6.2.6.12	Review and inspection procedures [PAF; SRR, PDR, CDR, QR, AR, ORR].
Q-80B-6.3.2.3	Review and inspection records [PAF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-6.3.2.3	Design standards [PAF; SRR]

- Q-80B-6.3.2.9-a Description of checks in the software product assurance plan [PAF; SRR, PDR]
- Q-80B-7.1.1 Software product assurance plan – assurance activities [PAF; SRR, PDR]
- Q-80B-7.1.4.2 Products metrics specification and justification in software product assurance plan [PAF; SRR, PDR]
- Q-80B-7.1.7 Products metrics specification and justification in software product assurance plan [PAF; SRR, PDR]
- Q-80B-7.2.2.3-a. Product quality requirements reflected in coding and design standards [PAF; SRR, PDR]
- Q-80B-7.3.2 Software product assurance report [PAF; SRR, PDR, CDR, QR, AR, ORR]

#### **A.9.4 PAF contents at PDR**

- Q-80B-5.4.1.1,
- Q-80B-5.4.1.2
- Q-80B-5.4.1.4 Software product assurance plan [PAF; SRR, PDR]
- Q-80B-5.4.1.3 Software product assurance plan [PAF; CDR, QR, AR, ORR]
- Q-80B-5.4.1.6 Compliance matrix [PAF; SRR, PDR]
- Q-80B-5.4.2.2 Assessment of the quality of software development process in the software product assurance report [PAF; SRR, PDR, CDR, QR, AR, ORR]
- Q-80B-5.4.2.3 Assessment of the quality of software product in the software product assurance report [PAF; SRR, PDR, CDR, QR, AR, ORR]
- Q-80B-5.4.2.4 Software product assurance report [PAF; SRR, PDR, CDR, QR, AR, ORR]
- Q-80B-5.4.5.3 Identification of the point in the software life cycle from which the nonconformance procedures apply – software product assurance plan [PAF; SRR, PDR]
- Q-80B-5.4.6.1,
- Q-80B-5.4.6.2 Software problem reporting procedures [PAF; PDR]
- Q-80B-5.6.3.3,
- Q-80B-5.6.3.4 Next level suppliers' software product assurance plan [PAF; PDR]
- Q-80B-5.7.6 Receiving inspection report [PAF; PDR, CDR, QR]
- Q-80B-5.8.3.1,
- Q-80B-5.8.3.2 Justification included or referenced in the product assurance file [PAF; PDR]
- Q-80B-5.8.3.5 Software product assurance report [PAF; SRR, PDR, CDR, QR, AR, ORR]
- Q-80B-6.1.1,
- Q-80B-6.1.5,
- Q-80B-6.1.6,

Q-80B-6.1.7,	
Q-80B-6.1.8,	
Q-80B-6.1.9	Reference to software development life cycle definition [PAF; SRR, PDR]
Q-80B-6.2.1.4	Identification of software plans, their interrelations and schedule for preparation [PAF; SRR, PDR].
Q-80B-6.2.1.6,	
Q-80B-6.2.1.7,	
Q-80B-6.2.1.8	Procedures and Standards [PAF; PDR]
Q-80B-6.2.2.2-a	Software criticality analysis report [PAF; SRR, PDR]
Q-80B-6.2.2.2-b	Input to the system safety data package [PAF; SRR, PDR]
Q-80B-6.2.2.3	List of critical software components [PAF; PDR]
Q-80B-6.2.3.1,	
Q-80B-6.2.3.3,	
Q-80B-6.2.3.4	Definition of measures and verification activities in software product assurance plan [PAF; PDR, CDR]
Q-80B-6.2.4.8-a,	
Q-80B-6.2.4.9,	
Q-80B-6.2.4.11	Identification and protection method or tool [PAF; SRR, PDR]
Q-80B-6.2.4.12-a	Labelling method [PAF; SRR, PDR].
Q-80B-6.2.5.1,	
Q-80B-6.2.5.2,	
Q-80B-6.2.5.3	Details of metrics in software product assurance plan [PAF; SRR, PDR]
Q-80B-6.2.5.4,	
Q-80B-6.2.5.5,	
Q-80B-6.2.5.6	Metrics reports in software product assurance reports [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-6.2.6.2-a	SPA reports [PAF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-6.2.6.8,	
Q-80B-6.2.6.9	Review and inspection procedures [PAF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-6.2.6.11,	
Q-80B-6.2.6.12	Review and inspection records [PAF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-6.3.2.3	Design standards [PAF; SRR, PDR]
Q-80B-6.3.2.4	Design and coding rules for numerical accuracy [PAF; PDR]
Q-80B-6.3.2.5,	
Q-80B-6.3.2.6,	
Q-80B-6.3.2.8	Results in software product assurance reports [PAF; PDR, CDR]
Q-80B-6.3.2.7	Description of checks in the software product assurance plan [PAF; PDR, CDR]

Q-80B-6.3.2.9-a	Description of checks in the software product assurance plan [PAF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-6.3.2.9-b	Results in software product assurance reports [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-6.3.3.1	Coding standards [PAF; PDR]
Q-80B-6.3.3.2,	
Q-80B-6.3.3.3,	
Q-80B-6.3.3.4	Coding standards and description of tools [PAF; PDR]
Q-80B-6.3.3.6	Description of measurements and tools [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-6.3.3.7	Description of measurements and tools [PAF; PDR]
Q-80B-6.3.4.1,	
Q-80B-6.3.4.2	Assurance activities for testing in the software product assurance plan [PAF; PDR, CDR]
Q-80B-7.1.1	Software product assurance plan – assurance activities [PAF; SRR, PDR]
Q-80B-7.1.3.1,	
Q-80B-7.1.3.2	Software quality models [PAF; PDR]
Q-80B-7.1.4.2	Products metrics specification and justification in software product assurance plan [PAF; SRR, PDR]
Q-80B-7.1.5	Report of the analysis and remedial actions in the software product assurance report [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-7.1.6	Report of the analysis and metrics in the software product assurance report [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-7.1.7	Product metrics specification and justification in software product assurance plan [PAF; SRR, PDR]
Q-80B-7.1.8	Report of the analysis and metrics in the software product assurance report [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-7.1.9	Report of the analysis and remedial actions in the software product assurance report [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-7.1.11	Report of the analysis of software behaviour in the software product assurance report [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-7.1.12	Report of the analysis and metrics in the software product assurance plan [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-7.1.13	Records of data collection analysis and results and actions for improvement [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-7.2.2.3-a	Product quality requirements reflected in coding and design standards [PAF; SRR, PDR]
Q-80B-7.3.2	Software product assurance report [PAF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-7.5.1	Procedures described or referenced in the software product assurance plan [PAF; PDR]
Q-80B-7.5.2	Marking described or referenced in the software product assurance plan [PAF; PDR]

### A.9.5 PAF contents at CDR

Q-80B-5.4.1.3	Software product assurance plan [PAF; CDR, QR, AR, ORR]
Q-80B-5.4.2.2	Assessment of the quality of software development process in the software product assurance report [PAF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-5.4.2.3	Assessment of the quality of software product in the software product assurance report [PAF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-5.4.2.4	Software product assurance report [PAF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-5.7.6	Receiving inspection report [PAF; PDR, CDR, QR]
Q-80B-5.8.3.5	Software product assurance report [PAF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-6.2.2.4-a	Software criticality analysis report [PAF; CDR, QR, AR, ORR]
Q-80B-6.2.2.4-b	Input to the system safety data package [PAF; CDR, QR, AR, ORR]
Q-80B-6.2.2.5	List of critical software components [PAF; CDR, QR, AR, ORR]
Q-80B-6.2.3.1, Q-80B-6.2.3.3, Q-80B-6.2.3.4	Definition of measures and verification activities in software product assurance plan [PAF; PDR, CDR]
Q-80B-6.2.5.4, Q-80B-6.2.5.5, Q-80B-6.2.5.6	Metrics reports in software product assurance reports [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-6.2.6.2-a	SPA reports [PAF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-6.2.6.5-a, Q-80B-6.2.6.6-a	SPA reports [PAF; CDR, QR, AR, ORR]
Q-80B-6.2.6.8, Q-80B-6.2.6.9	Review and inspection procedures [PAF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-6.2.6.11, Q-80B-6.2.6.12	Review and inspection records [PAF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-6.3.2.5, Q-80B-6.3.2.6, Q-80B-6.3.2.8	Results in software product assurance reports [PAF; PDR, CDR]
Q-80B-6.3.2.9-b	Results in software product assurance reports [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-6.3.3.6	Description of measurements and tools [PAF; PDR, CDR, QR, AR, ORR]



Q-80B-6.3.3.8	Description of measurements and synthesis in software product assurance reports [PAF; CDR, QR, AR, ORR]
Q-80B-6.3.4.1,	
Q-80B-6.3.4.2	Assurance activities for testing in the software product assurance plan [PAF; PDR, CDR]
Q-80B-6.3.4.3,	
Q-80B-6.3.4.5,	
Q-80B-6.3.4.6	Collected data and analysis of the results in the software product assurance report [PAF; CDR, QR, AR, ORR]
Q-80B-6.3.4.8,	
Q-80B-6.3.4.12,	
Q-80B-6.3.4.13,	
Q-80B-6.3.4.14	Statement of compliance with test plans and procedures [PAF; CDR, QR, AR, ORR]
Q-80B-6.3.4.30-a,	
Q-80B-6.3.4.31-a,	
Q-80B-6.3.4.32-a	SPA reports [PAF; CDR, QR, AR, OR]
Q-80B-7.1.5	Report of the analysis and remedial actions in the software product assurance report [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-7.1.6	Report of the analysis and metrics in the software product assurance report [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-7.1.8	Report of the analysis and metrics in the software product assurance report [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-7.1.9	Report of the analysis and remedial actions in the software product assurance report [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-7.1.11	Report of the analysis of software behaviour in the software product assurance report [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-7.1.12	Report of the analysis and metrics in the software product assurance plan [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-7.1.13	Records of data collection analysis and results and actions for improvement [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-7.3.2	Software product assurance report [PAF; SRR, PDR, CDR, QR, AR, ORR]

#### **A.9.6 PAF contents at QR**

Q-80B-5.4.1.3	Software product assurance plan [PAF; CDR, QR, AR, ORR]
Q-80B-5.4.2.2	Assessment of the quality of software development process in the software product assurance report [PAF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-5.4.2.3	Assessment of the quality of software product in the software product assurance report [PAF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-5.4.2.4	Software product assurance report [PAF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-5.7.6	Receiving inspection report [PAF;PDR, CDR, QR]



Q-80B-5.8.3.5	Software product assurance report [PAF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-6.2.2.4-a	Software criticality analysis report [PAF; CDR, QR, AR, ORR]
Q-80B-6.2.2.4-b	Input to the system safety data package [PAF; CDR, QR, AR, ORR]
Q-80B-6.2.2.5	List of critical software components [PAF; CDR, QR, AR, ORR]
Q-80B-6.2.5.4, Q-80B-6.2.5.5, Q-80B-6.2.5.6	Metrics reports in software product assurance reports [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-6.2.6.2-a	SPA reports [PAF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-6.2.6.5-a, Q-80B-6.2.6.6-a, Q-80B-6.2.6.8, Q-80B-6.2.6.9	Review and inspection procedures [PAF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-6.2.6.11, Q-80B-6.2.6.12	Review and inspection records [PAF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-6.3.2.9-b	Results in software product assurance reports [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-6.3.3.6	Description of measurements and tools [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-6.3.3.8	Description of measurements and synthesis in software product assurance reports [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-6.3.4.3, Q-80B-6.3.4.5, Q-80B-6.3.4.6	Collected data and analysis of the results in the software product assurance report [PAF; CDR, QR, AR, ORR]
Q-80B-6.3.4.8 , Q-80B-6.3.4.12, Q-80B-6.3.4.13, Q-80B-6.3.4.14	Statement of compliance with test plans and procedures [PAF; CDR, QR, AR, ORR]
Q-80B-6.3.4.30-a, Q-80B-6.3.4.31-a, Q-80B-6.3.4.32-a	SPA reports [PAF; CDR, QR, AR, OR]
Q-80B-7.1.5	Report of the analysis and remedial actions in the software product assurance report [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-7.1.6	Report of the analysis and metrics in the software product assurance report [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-7.1.8	Report of the analysis and metrics in the software product assurance report [PAF; PDR, CDR, QR, AR, ORR]

Q-80B-7.1.9	Report of the analysis and remedial actions in the software product assurance report [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-7.1.11	Report of the analysis of software behaviour in the software product assurance report [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-7.1.12	Report of the analysis and metrics in the software product assurance plan [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-7.1.13	Records of data collection analysis and results and actions for improvement [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-7.3.2	Software product assurance report [PAF; SRR, PDR, CDR, QR, AR, ORR]

#### **A.9.7 PAF contents at AR**

Q-80B-5.4.1.3	Software product assurance plan [PAF; CDR, QR, AR, ORR]
Q-80B-5.4.1.5	Quality measures for the operations and maintenance processes in the software product assurance plan [PAF; AR]
Q-80B-5.4.2.2	Assessment of the quality of software development process in the software product assurance report [PAF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-5.4.2.3	Assessment of the quality of software product in the software product assurance report [PAF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-5.4.2.4	Software product assurance report [PAF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-5.8.3.5	Software product assurance report [PAF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-6.2.2.4-a	Software criticality analysis report [PAF; CDR, QR, AR, ORR]
Q-80B-6.2.2.4-b	Input to the system safety data package [PAF; CDR, QR, AR, ORR]
Q-80B-6.2.2.5	List of critical software components [PAF; CDR, QR, AR, ORR]
Q-80B-6.2.5.4, Q-80B-6.2.5.5, Q-80B-6.2.5.6	Metrics reports in software product assurance reports [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-6.2.6.2-a	SPA reports [PAF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-6.2.6.5-a, Q-80B-6.2.6.6-a,	SPA reports [PAF; CDR, QR, AR, ORR]
Q-80B-6.2.6.8, Q-80B-6.2.6.9	Review and inspection procedures [PAF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-6.2.6.11, Q-80B-6.2.6.12	Review and inspection records [PAF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-6.3.2.9-b	Results in software product assurance reports [PAF; PDR, CDR, QR, AR, ORR]

Q-80B-6.3.3.6	Description of measurements and tools [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-6.3.3.8	Description of measurements and synthesis in software product assurance reports [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-6.3.4.3, Q-80B-6.3.4.5, Q-80B-6.3.4.6	Collected data and analysis of the results in the software product assurance report [PAF; CDR, QR, AR, ORR]
Q-80B-6.3.4.8, Q-80B-6.3.4.12, Q-80B-6.3.4.13, Q-80B-6.3.4.14	Statement of compliance with test plans and procedures [PAF; CDR, QR, AR, ORR]
Q-80B-6.3.4.30-a, Q-80B-6.3.4.31-a, Q-80B-6.3.4.32-a	SPA reports [PAF; CDR, QR, AR, OR]
Q-80B-6.3.6.2	Contribution to the validation of the operational requirements [PAF; AR]
Q-80B-7.1.5	Report of the analysis and remedial actions in the software product assurance report [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-7.1.6	Report of the analysis and metrics in the software product assurance report [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-7.1.8	Report of the analysis and metrics in the software product assurance report [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-7.1.9	Report of the analysis and remedial actions in the software product assurance report [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-7.1.11	Report of the analysis of software behaviour in the software product assurance report [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-7.1.12	Report of the analysis and metrics in the software product assurance plan [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-7.1.13	Records of data collection analysis and results and actions for improvement [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-7.3.2	Software product assurance report [PAF; SRR, PDR, CDR, QR, AR, ORR]

#### **A.9.8 PAF contents at ORR**

Q-80B-5.4.1.3	Software product assurance plan [PAF; CDR, QR, AR, ORR]
Q-80B-5.4.2.2	Assessment of the quality of software development process in the software product assurance report [PAF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-5.4.2.3	Assessment of the quality of software product in the software product assurance report [PAF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-5.4.2.4	Software product assurance report [PAF; SRR, PDR, CDR, QR, AR, ORR]

Q-80B-5.8.3.5	Software product assurance report [PAF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-6.2.2.4-a	Software criticality analysis report [PAF; CDR, QR, AR, ORR]
Q-80B-6.2.2.4-b	Input to the system safety data package [PAF; CDR, QR, AR, ORR]
Q-80B-6.2.2.5	List of critical software components [PAF; CDR, QR, AR, ORR]
Q-80B-6.2.5.4, Q-80B-6.2.5.5, Q-80B-6.2.5.6	Metrics reports in software product assurance reports [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-6.2.6.2-a	SPA reports [PAF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-6.2.6.5-a, Q-80B-6.2.6.6-a,	SPA reports [PAF; CDR, QR, AR, ORR]
Q-80B-6.2.6.8, Q-80B-6.2.6.9	Review and inspection procedures [PAF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-6.2.6.11, Q-80B-6.2.6.12	Review and inspection records [PAF; SRR, PDR, CDR, QR, AR, ORR]
Q-80B-6.3.2.9-b	Results in software product assurance reports [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-6.3.3.6	Description of measurements and tools [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-6.3.3.8	Description of measurements and synthesis in software product assurance reports [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-6.3.4.3, Q-80B-6.3.4.5, Q-80B-6.3.4.6	Collected data and analysis of the results in the software product assurance report [PAF; CDR, QR, AR, ORR]
Q-80B-6.3.4.8, Q-80B-6.3.4.12, Q-80B-6.3.4.13, Q-80B-6.3.4.14	Statement of compliance with test plans and procedures [PAF; CDR, QR, AR, ORR]
Q-80B-6.3.6.3	Input to product assurance plan for systems operation [PAF; ORR]
Q-80B-7.1.5	Report of the analysis and remedial actions in the software product assurance report [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-7.1.6	Report of the analysis and metrics in the software product assurance report [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-7.1.8	Report of the analysis and metrics in the software product assurance report [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-7.1.9	Report of the analysis and remedial actions in the software product assurance report [PAF; PDR, CDR, QR, AR, ORR]

Q-80B-7.1.11	Report of the analysis of software behaviour in the software product assurance report [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-7.1.12	Report of the analysis and metrics in the software product assurance plan [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-7.1.13	Records of data collection analysis and results and actions for improvement [PAF; PDR, CDR, QR, AR, ORR]
Q-80B-7.3.2	Software product assurance report [PAF; SRR, PDR, CDR, QR, AR, ORR]

## A.10 System level documentation

### A.10.1 General

The system level documentation is governed by the ECSS system engineering standard. The relevant input for software elements at system level are found in subclause 5.2. In the special case where the customer is himself a software supplier (a product consisting solely of software) for the next higher level in the system product tree, the *customer* becomes a *supplier* at that level and the requirements of this current standard are applied recursively for that case.

### A.10.2 DDF contents at SRR

E-40B-5.2.3.1a System design [DDF-system level; SRR]

### A.10.3 DJF contents at SRR

E-40B-5.2.4.4 Requirements justification [DJF-system level; SRR]

E-40B-5.2.3.1b System design to system requirements conformance [DJF-system level; SRR]

E-40B-5.2.3.1c System requirements to system design traceability [DJF-system level; SRR]

*(This page is intentionally left blank)*

## Annex B (informative)

### References to other ECSS Standards

Referenced ECSS Standard	clause / page(s)
ECSS-E-00	cl4 / 19, 28
ECSS-E-10	cl4 / 23, 28
ECSS-E-10A	cl5 / 34
ECSS-E-40-01	cl5 / 31
ECSS-E-40-04	cl5 / 38
ECSS-E-70	cl4 / 25, 30
ECSS-M	cl4 / 26
ECSS-M-00	cl4 / 27
ECSS-M-00B	cl5 / 55
ECSS-M-00-02	cl4 / 30
ECSS-M-00-03	cl4 / 27
ECSS-M-10	cl4 / 27
ECSS-M-20	cl4 / 19, 27; cl5 / 51, 56
ECSS-M-30	cl4 / 19, 27; cl5 / 39, 42, 61
ECSS-M-40	cl4 / 28; cl5 / 43, 61, 62, 64
ECSS-M-50	cl4 / 28
ECSS-M-60	cl4 / 28
ECSS-M-70	cl4 / 28
ECSS-Q-20B	cl5 / 54
ECSS-Q-80	cl4 / 18, 26; cl5 / 38
ECSS-Q-80B	cl5 / 33, 42, 46, 47, 50, 51, 54, 55, 56; 59; 62; 64

*(This page is intentionally left blank)*



## Annex C (informative)

---

### Tailoring guidelines

#### C.1 Introduction

The ECSS family of standards is intended to be tailored for each individual project. This Standard lists exhaustively the requirements for the best practices in space software engineering. The purpose of this clause is to give the customer general guidelines on tailoring this standard for a specific project.

The goal of the tailoring is to select, modify or add adequately requirements in order to reach the optimized ratio quality and the actual project peculiarities. The technical, management and operational factors discussed in the following subclauses provide for the examples of the software project peculiarities to be considered, to perform tailoring of this Standard.

#### C.2 How to tailor

The first step is therefore to understand the requested level of quality for the project, which starts with the characterisation of the project, the identification of the needed processes, and the characterisation of the product.

ECSS-M-00-02 provides for indication on the general way to tailor an ECSS standard, in particular the tailoring process and the tailoring templates. The templates are generic and deserve a more concrete description of the so-called programmatic and technical factors.

For space software, there are some examples where the full application of this Standard can be tailored and then refined on the basis of software project details (the tailoring factors). The main influencing factors are the following:

- Technical factors:
  - novelty of the domain of application;
  - complexity of the software and the system;
  - criticality level;
  - size of the software;
  - reusability required of the software being developed;
  - interface to system development projects;
  - degree of use of COTS or existing software;
  - maturity of the COTS and completeness or stability of the user requirements.

- Operational factors:
  - type of application (e.g. platform, payload, and experiment);
  - number of potential users of the software;
  - criticality of the software as measured by the consequences of its failure;
  - expected lifetime of the software;
  - number of sites where the software is used;
  - operation, maintenance, migration and retirement constraints.
- Management factors:
  - amount of time and effort required to develop the software;
  - budget requirements for implementing and operating the software;
  - accepted risk level for the project;
  - type of life cycle;
  - schedule requirements for delivering the software;
  - number of people required to develop, operate and maintain the software;
  - complexity of the organization;
  - experience of the supplier;
  - financial resource.

For each particular project additional factors may be used.

The tailoring can be made during a short discussion between the software engineering engineer and the software project manager. The software engineering engineer first asks a set of questions to the project manager, in order to set up the scope of the tailoring (i.e. the characteristics of the project that influence the selection or not of each requirements). Examples of questions are:

- Who are the customer, the supplier, the user, the maintainer, and the operator? Does the customer intend to delegate some tasks to the supplier?
- Where is the complexity of the project, in the requirements or in the design?
- What level of validation is necessary? Should the product be perfect at delivery, or is some room allowed for the user to participate to the tests, or is it a prototype that will be dropped later on (or reused in the next phase)?
- What level of verification is necessary? Is it necessary to verify the requirements, or the code, or the test definition?
- What visibility into the design is wished? Does the project manager want to know everything on the detailed design and unit test, or does he trust the supplier for a part of the life cycle?
- Consequently, what are the necessary reviews to be selected into the project? Is it acceptable to merge some of them (as QR and AR, or SRR and PDR) or to waive others (as CDR or DDR)?
- How much are COTS involved? Is the project an assembly of COTS products where the COTS acceptance and integration shall have the emphasis?
- Is the software critical? Is it included into an hardware environment?
- How is the maintenance organized? Is it included fully in the current contract, or is the maintenance limited to the guarantee period?

Then the requirements in clauses 5 and 6 are reviewed and made or not applicable in a table.

The tailoring of this Standard results in a short document including the project characteristics (as a justification for the tailoring) and the tailoring table.

Note that several subclauses are anyhow mandatory for any project, e.g. the production of a minimum set of software requirements, a PDR to review them, and the production of the code.

### **C.3 Who tailors?**

The tailoring of this Standard is implicitly a task of the Customer. When preparing the Invitation to Tender, the Customer proposes a tailored version of the standard as an indication of the level of software engineering to be applied for the project. However, some tailoring factors (such as criticality, detailed design complexity) may only be known after the grant of the contract. The Supplier is also part of the tailoring process and the resulting document is baselined in the RB (at SRR). The Customer can also subcontract the tailoring to the Supplier, then review and accept the tailored version.

*(This page is intentionally left blank)*

---

## Bibliography

ECSS-E-00	Space engineering — Policy and principles
ECSS-E-10-01 <sup>1)</sup>	Space engineering — Standard practice for interface management
ECSS-E-40-01 <sup>1)</sup>	Space engineering — Space segment software
ECSS-E-40-03 <sup>1)</sup>	Space engineering — Ground segment software
ECSS-E-40-04 <sup>1)</sup>	Space engineering — Software life cycles
ECSS-E-70	Space engineering - Grounds systems and operations
ECSS-M-00-02A	Space project management — Tailoring of space standards
ECSS-M-00-03	Space project management — Risk management
ECSS-M-50	Space project management — Information/documentation management
ECSS-M-60	Space project management — Cost and schedule management
ECSS-M-70	Space project management — Integrated logistic support
ECSS-Q-30	Space product assurance — Dependability
ECSS-Q-40	Space product assurance — Safety
ISO 9000:2000	Quality management systems — Fundamentals and vocabulary
ISO 9126-1:2001	Software engineering — Product quality — Part 1: Quality model
ISO/IEC 2382-20:1990	Information technology — Vocabulary — Part 20: System development
ISO/IEC 12207:1995	Information technology — Software life cycle processes
IEEE 610.12-1990	Standard glossary of software engineering terminology
IEEE 1062-1993	Recommended practice for software acquisition

---

1) To be published.

*(This page is intentionally left blank)*

## ECSS Document Improvement Proposal

<b>1. Document I.D.</b> ECSS-E-40 Part 1B	<b>2. Document date</b> 28 November 2003	<b>3. Document title</b> Software - Part 1: Principles and requirements
<b>4. Recommended improvement</b> (identify clauses, subclauses and include modified text or graphic, attach pages as necessary)		
<b>5. Reason for recommendation</b>		
<b>6. Originator of recommendation</b>		
Name:	Organization:	
Address:	Phone: Fax: e-mail:	<b>7. Date of submission:</b>
<b>8. Send to ECSS Secretariat</b>		
Name: W. Kriedte ESA-TOS/QR	Address: ESTEC, P.O. Box 299 2200 AG Noordwijk The Netherlands	Phone: +31-71-565-3952 Fax: +31-71-565-6839 e-mail: Werner.Kriedte@esa.int

**Note:** The originator of the submission should complete items 4, 5, 6 and 7.

An electronic version of this form is available in the ECSS website at: <http://www.ecss.nl/>  
At the website, select "Standards" - "ECSS forms" - "ECSS Document Improvement Proposal"

*(This page is intentionally left blank)*