ECSS-E-40 Part 2B
31 March 2005

Space engineering

Software — Part 2: Document requirements definitions (DRDs)

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

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

ISSN: 1028-396X

Price: € 20

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.

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.

This Standard is divided in two parts:

Part 1, which defines the principles and requirements applicable to space software engineering

Part 2, which defines the content of the DRDs which are specifically referenced in ECSS-E-40 Part 1 and in ECSS-Q-80.

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

(This page is intentionally left blank)

Contents

Tables

1

Scope

Part 2 of ECSS-E-40 “Space engineering - Software” defines the content of the document requirements definitions (DRDs) for space software product, in order to organize into documents all the expected outputs of the requirements contained in of ECSS-E-40 Part 1 “Space engineering - Software” and in ECSS-Q-80 “Space product assurance - Software product assurance”.

(This page is intentionally left blank)

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-40 Part 1B	Space engineering — Software — Part 1: Principles and requirements
ECSS-Q-80B	Space product assurance — Software product assurance

(This page is intentionally left blank)

3

Terms, definitions and abbreviated terms

3.1 Terms and definitions

For the purposes of this document, the terms and definitions given in ECSS-P-001, ECSS-E-40 Part 1, ECSS-Q-80 and the following apply.

3.1.1
test
formal process of exercising or putting to trial a system or item by manual or automatic means to identify differences between specified, expected and actual results
[ECSS-P-001B]

3.1.2
test case
set of test inputs, execution conditions and expected results developed for a particular objective such as to exercise a particular program path or to verify compliance with a specified requirement

3.1.3
test design
documentation specifying the details of the test approach for a software feature or combination of software features and identifying associated tests

3.1.4
test procedure
detailed instructions for the set up, operation and evaluation of the results for a given test

3.1.5
test script
file containing a set of commands or instructions written in native format (computer or tool processable) in order to automate the execution of one or a combination of test procedures (and the associated evaluation of the results)

3.2 Abbreviated terms

The abbreviated terms defined in ECSS-P-001 and ECSS-E-40 Part 1, and the following apply:

CCN	contractual change notice
DRD	document requirements definition
SCF	software configuration file
SCR	software change request
SDD	software design document
SDP	software development plan
SPAP	software product assurance plan
SRelD	software release document
SRF	software reuse file
SRS	software requirements specification
SSS	software system specification
SUITP	software unit/integration test plan
SValP	software validation plan
SVerP	software verification plan
SVTS	software validation testing specifications
SW&D	software waiver and deviation
w.r.t.	with respect to

4

Document requirements definitions (DRD) list

ECSS Standards specify the production and use of project documents. Document requirements definitions are defined to control the content of the project documents.

Document requirements definitions serve to ensure:

a. completeness and consistency of information within documents,

b. that the information contained in a document conforms to its defined scope, and correctly implements its interfaces with other documents, and

c. that portions of a document can be generated or maintained by separate organizational groups and seamlessly integrated into a coherent whole.

Table 1 lists and gives a summary of the DRDs that are defined in the annexes of this Standard and called up in ECSS-E-40 Part 1 and ECSS-Q-80.

Table 2 lists all the documents called up in ECSS-E-40 Part 1 and ECSS-Q-80, and describes whether or not they are part of a DRD, which reviews and milestones they are associated with, and which files they are part of.

The tailoring principles described in ECSS-E-40 Part 1 are also valid for ECSS-E-40 Part 2.

Table 1: ECSS-E-40 and ECSS-Q-80 DRD list

DRD ID	DRD title	DRD description and purpose	Related file or folder	Delivered at (review)
ECSS-E-40 Part 2B Annex A	Software system specification (SSS)	The software system specification contains the customer's requirements. It is generated by the system engineering processes related to software. It is the highest level description of the software and, together with the IRD provides the criteria used to validate and accept the software. This DRD can be integrated into upper level documentation.	RB	SRR
ECSS-E-40 Part 2B Annex B	Software requirements specification (SRS)	The software requirements specification describes all the requirements applicable to the software item.	TS	PDR
ECSS-E-40 Part 2B Annex C	Software design document (SDD)	The software design document describes the software architectural design and the software detailed design. Internal interfaces design is also included in this document.	DDF	PDR and CDR
ECSS-E-40 Part 2B Annex D	Software release document (SRelD)	The purpose of this DRD is to describe the software release document, in the frame of project using ECSS standards. It describes a given software version in terms of known problems, limitations or restrictions with respect to its approved baseline.	DDF	QR and AR
ECSS-E-40 Part 2B Annex E	Software [unit/integration] test plan (SUITP)	The purpose of this DRD is to describe the tests plans, in the frame of project using ECSS standards. This DRD provides a unique template for unit and integration testing to be instantiated with respect to the software test plans specified in the document requirement list, either for a software unit test plan, either for a software integration test plan, or for both.	DJF	CDR
ECSS-E-40 Part 2B Annex F	Software validation testing specification (SVTS)	The purpose of this DRD is to describe the testing specifications in the frame of project using ECSS standards. The addressed software validation testing specification are the software validation testing specification with respect to the technical specification (TS) and the software validation testing specification with respect to the requirements baseline (RB).	DJF	CDR, QR

Table 1: ECSS-E-40 and ECSS-Q-80 DRD list *(continued)*

DRD ID	DRD title	DRD description and purpose	Related file or folder	Delivered at (review)
ECSS-E-40 Part 2B Annex G	Software verification plan (SVerP)	The software verification plan DRD is produced in order to describe the software verification approach and the organizational aspects of the verification activities to be executed.	DJF	PDR
ECSS-E-40 Part 2B Annex H	Software validation plan (SValP)	The software validation plan describes the approach to the implementation of the validation process for a software product. The software validation plan provides for the definition of organizational aspects and management approach to the implementation of the validation tasks.	DJF	PDR
ECSS-E-40 Part 2B Annex I	Software reuse file (SRF)	The purpose of this DRD is to describe the software reuse file in the frame of ECSS based project. The software reuse file is a constituent of the design justification file (DJF) to be developed in the course of a software development project. It documents the output of analysis to be performed on existing software intended to be reused. The global aim of the software reuse file is to document all the information used to decide about the reuse (or not) of existing software and specific action plan related to identified risks.	DJF	SRR, PDR, CDR
ECSS-E-40 Part 2B Annex J	Software development plan (SDP)	The software development plan describes the established management and development approach for the software items to be defined by a software supplier to set up a software project in accordance with the customer requirements.	MGT	SRR and PDR
ECSS-E-40 Part 2B Annex K	Software product assurance plan (SPAP)	The software product assurance plan provides information on the organizational aspects and the technical approach to the execution of the software product assurance programme.	PAF	SRR and PDR

Table 2: ECSS-E-40 and ECSS-Q-80 Document requirement list (DRL)

Related file	DRL item (e.g. Plan, document, file, report, form, matrix)	DRL item having a DRD	SRR	PDR	DDR (see NOTE 6)	CDR	QR	AR	ORR
	Software system specification (SSS)	ECSS-E-40 Part 2B Annex A	✓						
	Software interface requirements document	-	✓						
	System partition with definition of items	-	✓						
	System configuration items list (see NOTE 1)	-	✓						
	Interface management procedures	-	✓						
TS	Software requirements specification (SRS)	ECSS-E-40 Part 2B Annex B		✓					
	Software interface control document	-		✓	✓	✓			

Table 2: ECSS-E-40 and ECSS-Q-80 Document requirement list (DRL) *(continued)*

Related file	DRL item (e.g. Plan, document, file, report, form, matrix)	DRL item having a DRD	SRR	PDR	DDR (see NOTE 6)	CDR	QR	AR	ORR
DDF	Software design document (SDD)	ECSS-E-40 Part 2B Annex C		✔	✔	✔			
	Software configuration file (SCF)	ECSS-M-40B Annex F		✔	✔	✔	✔	✔	✔
	Software release document (SRelD)	ECSS-E-40 Part 2B Annex D					✔	✔	
	Software user manual	-			✔	✔	✔	✔	
	Software source code	-				✔			
	Software delivery	-					✔	✔	✔
	Training material	-					✔		

Table 2: ECSS-E-40 and ECSS-Q-80 Document requirement list (DRL) *(continued)*

Related file	DRL item (e.g. Plan, document, file, report, form, matrix)	DRL item having a DRD	SRR	PDR	DDR (see NOTE 6)	CDR	QR	AR	ORR
DJF	Software verification plan (SVerP)	ECSS-E-40 Part 2B Annex G		✓					
	Software validation plan (SValP)	ECSS-E-40 Part 2B Annex H		✓					
	Independent software verification & validation plan	-		✓					
	Software integration test plan	ECSS-E-40 Part 2B Annex E		✓	✓	✓			
	Software unit test plan	ECSS-E-40 Part 2B Annex E			✓	✓			
	Software validation testing specification (SVTS) with respect to TS	ECSS-E-40 Part 2B Annex F				✓			
	Software validation testing specification (SVTS) with respect to RB	ECSS-E-40 Part 2B Annex F					✓		
	Acceptance test plan	-						✓	
	Installation plan	-						✓	
	Software unit test report	-				✓			
	Software integration test report	-				✓			
	Software validation test report with respect to TS	-				✓			
	Software validation test report with respect to RB	-					✓	✓	

Table 2: ECSS-E-40 and ECSS-Q-80 Document requirement list (DRL) *(continued)*

Related file	DRL item (e.g. Plan, document, file, report, form, matrix)	DRL item having a DRD	SRR	PDR	DDR (see NOTE 6)	CDR	QR	AR	ORR
DJF	Acceptance testing documentation							✓	
	Acceptance test report	-						✓	
	Installation report	-						✓	
	(Analyses, Inspection and RoD) verification report with respect to TS	-				✓			
	(Analyses, Inspection and RoD) verification report with respect to RB	-					✓		
	Software traceability matrices (see NOTE 5)	ECSS-E-40 Part 2B Annexes A, B, C, E, F	✓	✓	✓	✓	✓	✓	
	Traceability to system partitioning	-	✓						
	Independent software verification & validation report	-		✓	✓	✓	✓	✓	✓
	Software reuse file (SRF)	ECSS-E-40 Part 2B Annex I	✓	✓	✓	✓			
	Software budget report	-		✓	✓	✓	✓	✓	
	Numerical accuracy analysis			✓		✓	✓		
	Schedulability analyses	-		✓	✓	✓			
	Software behaviour verification	-		✓	✓	✓			

Table 2: ECSS-E-40 and ECSS-Q-80 Document requirement list (DRL) *(continued)*

Related file	DRL item (e.g. Plan, document, file, report, form, matrix)	DRL item having a DRD	SRR	PDR	DDR (see NOTE 6)	CDR	QR	AR	ORR
DJF	Software requirements verification report	-		✓					
	Software architectural design & interfaces verification report	-		✓					
	Software detailed design verification report	-			✓	✓			
	Software code verification report	-				✓			
	Software documentation verification report	-		✓		✓	✓		
	Software integration verification report	-				✓			
	Validation report evaluation with respect to TS	-				✓			
	Validation report evaluation with respect to RB	-					✓	✓	
	Software design and test evaluation report	-					✓		
	Testing feasibility report	-				✓			
	Problems and nonconformance report	-				✓	✓	✓	
	Milestones and technical review reports	-	✓	✓	✓	✓	✓	✓	✓
	Software acceptance data package (see NOTE 2)	-						✓	
	Procured software component list	-	✓	✓					

Table 2: ECSS-E-40 and ECSS-Q-80 Document requirement list (DRL) *(continued)*

Related file	DRL item (e.g. Plan, document, file, report, form, matrix)	DRL item having a DRD	SRR	PDR	DDR (see NOTE 6)	CDR	QR	AR	ORR
MGT	Software development plan (SDP)	ECSS-E-40 Part 2B Annex J	✓	✓					
	Software configuration management plan (see NOTE 3)	-	✓	✓					
	Training plan	-	✓						
	Customer approvals of documents (see NOTE 4)	-	✓	✓	✓	✓	✓	✓	
MF	Maintenance plan	-					✓		
	Maintenance records	-					✓	✓	✓
	PR and NCR - Modification analysis report - Problem analysis report - Modification identification	-							
	Migration plan	-							
	Retirement plan	-							
OP	Operational plan	-							✓
	Operational testing results	-							✓

Table 2: ECSS-E-40 and ECSS-Q-80 Document requirement list (DRL) *(continued)*

Related file	DRL item (e.g. Plan, document, file, report, form, matrix)	DRL item having a DRD	SRR	PDR	DDR (see NOTE 6)	CDR	QR	AR	ORR
PAF	Software product assurance plan (SPAP)	ECSS-E-40 Part 2B Annex K	✔	✔					
	Records of training and experience								
	Compliance matrix to the applicable software product assurance requirements	-	✔	✔					
	Software product assurance requirements for suppliers	-	✔						
	Software criticality analyses report	-	✔	✔	✔	✔	✔	✔	✔
	List of critical software components	-		✔	✔	✔	✔	✔	✔
	Software product assurance report	-	✔	✔	✔	✔	✔	✔	✔
	Audit plan	-	✔						
	Software process assessment plan	-							
	Software process assessment records	-							

Table 2: ECSS-E-40 and ECSS-Q-80 Document requirement list (DRL) *(continued)*

Related file	DRL item (e.g. Plan, document, file, report, form, matrix)	DRL item having a DRD	SRR	PDR	DDR (see NOTE 6)	CDR	QR	AR	ORR
System level	System design	-	✓						
	System design to system requirements conformance	-	✓						
	System requirements to system design traceability	-	✓						

NOTE 1 The system configuration items list is not issued by the software supplier (except if this supplier is also the supplier of the system in which the software is integrated).

NOTE 2 The software acceptance data package definition is in line with the notion of end item data package (EIDP) defined in ECSS-Q-20B.

NOTE 3 The software configuration management plan is not a formal output of the ECSS-E-40 Part 1B and the ECSS-Q-80B Standards but is listed here for the sake of completeness as a mandatory element of the management file, in accordance with ECSS-M-30.

NOTE 4 The customer approvals of documents is not a document but a proof that all the deliverable documents to be produced have been approved by the supplier (it can be e.g. a signature or a form).

NOTE 5 The software traceability matrices are included in the DRDs (i.e. software requirements to software system level requirements are included in the SRS), but they can also be part of a single document or database report.

NOTE 6 The detailed design review is held only for flight software.

(This page is intentionally left blank)

Annex A (normative)

Software system specification (SSS) DRD

A.1 DRD identification

A.1.1 Requirement identification and source document

The software system specification (SSS) document is called from the normative provisions summarized in Table A-1.

Table A-1: SSS traceability to ECSS-E-40 Part 1 and ECSS-Q-80 subclauses

ECSS Standard	Subclauses
ECSS-E-40 Part 1B	5.2.2.1 expected output a
	5.2.2.1 expected output c
	5.2.2.1 expected output d
	5.2.2.1 expected output e
	5.2.2.2
	5.2.2.4
	5.2.4.2
	5.2.4.3 expected output a
	5.2.4.3 expected output b
	5.2.5.1a
	5.2.5.1b
	5.2.5.1c
	5.2.5.1d
	5.2.5.4
	5.2.5.5
	5.2.5.6
	5.2.6.2
	5.2.7.1
	5.2.7.2
	5.3.5.1
ECSS-Q-80B	6.2.2.1

A.1.2 Purpose and objective

The software system specification contains the customer's requirements. It is generated by the system engineering processes related to software (see ECSS-E-40 Part 1). It is the highest level description of the software and, together with the interface requirements document, provides the criteria that are used to validate and accept the software.

The information about traceability to high-level requirements can be in the software system specification or in the requirements traceability in the design justification file. In either case a cross-reference is done.

The software system specification can be produced as a standalone document or as part of a system-level specification document. It can be included, for example, in the technical specification introduced by ECSS-E-10 Part 6. If produced as a standalone document, the present DRD applies, else the DRD described in ECSS-E-10 Part 6A for the establishment of a functional and technical specification applies.

The software system specification is a major component of the requirements baseline and is the primary input for the system requirements review (SRR)

A.2 Expected response

A.2.1 Response identification

The requirements for project identification contained in ECSS-M-50 shall be applied to the SSS.

A.2.2 Scope and content

The SSS shall provide the information presented in the following sections:

<1> Introduction

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

<2> Applicable and reference documents

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

<3> Terms, definitions and abbreviated terms

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

<4> General description

a. Product perspective

1. The SSS shall describe the product in perspective with other related systems.
2. If the product is to replace an existing system, the system shall be described and referenced.

b. General capabilities

The SSS shall describe the main capabilities to be supported by the software and the reasons to include these capabilities.

NOTE Reference to state and mode of the system can be made.

c. General constraints

The SSS shall describe any item that limits the supplier's options for designing and developing the software.

d. Operational environment

1. The SSS shall describe the software operational environment.
2. Context diagrams may support this narrative description to summarize external interfaces and system block diagrams to show how the activity fits within the larger system.
3. The nature of exchanges with external systems should be listed.
4. If a system specification defines a product that is a component of a parent system or project, then the SSS should list the activities that are supported by external systems.
5. References to the interface control documents that define the external interfaces with the other systems shall be provided
6. The computer infrastructure to be used shall be also described.

e. Assumptions and dependencies

The SSS shall list the assumptions that the specific requirements are based on.

NOTE 1 Risks analysis is used to identify assumptions that cannot prove to be valid.

NOTE 2 A constraint requirement, for example, can specify an interface with a system which does not exist.

NOTE 3 If the production of the system does not occur when expected, this system specification can change.

<5> Specific requirements

a. General

The following provisions apply to the system specific requirements listed in the SSS, as specified in b. to k. below:

1. Each requirement shall be uniquely identified.
2. The trace of each requirement derived from higher level documentation to the applicable higher level requirements shall be stated.

b. Capabilities requirements

1. The SSS shall list the requirements specifying the system behaviour and its associated performances.
2. Description of capability requirements shall be organized by capability type.

NOTE For example, requirements can be organized per controlled subsystem (e.g. AOCS, power, and thermal).

c. System interface requirements

1. The SSS shall list any interface requirements imposed on the system.
2. Requirements related to:
   (a) communication interfaces,
   (b) hardware interfaces,
   (c) software interfaces,
   (d) MMI,

   shall be either listed in the SSS or referenced in the IRD.

d. Adaptation requirements

The SSS shall list the data that can vary according to operational needs and any site-dependant data.

e. Computer resource requirements

1. Computer hardware resource requirements

The SSS shall list the requirements on the computer hardware to be used.

2. Computer hardware resources utilization requirements

The SSS shall list the requirements on the computer resource utilization (e.g. processor capacity and memory capacity) available for the software item (e.g. sizing and timing).

3. Computer software resource requirements

The SSS shall list requirements on the software items to be used by or incorporated into the system (or constituent software product) (e.g. a specific real time operating system)

f. Safety requirements

The SSS shall list the safety requirements applicable to the system.

g. Reliability requirements

The SSS shall list the reliability requirements applicable to the system.

h. Quality requirements

The SSS shall list the quality requirements applicable to the system (e.g. usability, reusability, and portability).

i. Design requirements and constraints

1. The SSS shall include the requirements which constraint the design and production of the system.
2. At least, the following type of requirements shall be included:
   (a) constraints on the software architecture,
   (b) utilization of standards,
   (c) utilization of existing components,
   (d) utilization of customer furnished components or COTS components,
   (e) utilization of design standard,
   (f) utilization of data standard,
   (g) utilization of a specific programming language,
   (h) utilization of specific naming convention,
   (i) flexibility and expansion, and
   (j) utilization of MMI standards.

j. Software operations requirements

The SSS shall include the system requirements for the operation of the software.

k. Software maintenance requirements

The SSS shall include the system requirements for the maintenance of the software.

<6> Validation approach and requirements

a. Validation approach

The SSS shall summarize the validation approach to be utilized to validate all the requirements stated in <5> and to ensure the requirements are met.

b. Validation requirements

1. The SSS shall describe the validation requirements specified to support the demonstration that software requirements are met.
2. For each of the identified requirements in <5>, a validation method shall be included.
3. If a given requirement need not be taken into account for the software validation against the requirements baseline, then it should be clearly identified.

NOTE A matrix (requirements to validation method correlation table) can be used to state the validation methods applicable to each requirement. This information can be further split into an information for validation requirements and an information for acceptance requirements, depending upon the project needs.

<7> Traceability

a. The SSS shall report the traceability matrices

1. from the upper level specification requirements to the requirements contained in <5>, and
2. from the requirements contained in <5> to the upper level applicable specification.

b. In case the information specified in a. is separately provided in the DJF, reference to this documentation shall be clearly stated.

(This page is intentionally left blank)

Annex B (normative)

Software requirements specification (SRS) DRD

B.1 DRD identification

B.1.1 Requirement identification and source document

The software requirements specification (SRS) document is called from the normative provisions summarized in Table B-1.

NOTE Apart from potential tailoring of ECSS-E-40 Part 1 and ECSS-Q-80 to be performed according to project specificity, these standards specify that the software requirements specification is provided at the PDR.

B.1.2 Purpose and objective

The software requirements specification is a major constituent of the technical specification (TS).

The software requirements specification describes the functional and non functional requirements applicable to the software item.

B.2 Expected response

B.2.1 Response identification

The requirements for project identification contained in ECSS-M-50 shall be applied to the SRS.

B.2.2 Scope and content

The SRS shall provide the information presented in the following sections:

<1> **Introduction**

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

<2> **Applicable and reference documents**

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

Table B-1: SRS traceability to ECSS-E-40 Part 1 and ECSS-Q-80 subclauses

ECSS Standard	Subclauses
ECSS-E-40 Part 1B	5.3.2.7 expected output a
	5.4.2.1 expected output a
	5.4.2.1 expected output b
	5.4.2.1 expected output c
	5.4.2.1 expected output d
	5.4.2.1 expected output e
	5.4.2.2 expected output a
	5.4.2.3
	5.4.2.4
	5.4.2.5
	5.4.2.6a
	5.4.3.11
	5.8.3.1 expected output a
ECSS-Q-80B	6.3.1.1
	6.3.1.3
	6.3.1.4
	7.1.2
	7.1.3.3
	7.2.1.1
	7.2.1.2
	7.2.1.3
	7.2.1.4

<3> Terms, definitions and abbreviated terms

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

<4> Software overview

a. Function and purpose

The SRS shall describe the purpose of the product.

b. Environmental considerations

The SRS shall summarize:

1. the physical environment of the target system;
2. the hardware environment in the target system;
3. the operating environment in the target system;
4. the hardware environment in the development system;
5. the operating environment in the development system.

c. Relation to other systems

1. The SRS shall describe in detail the product's relationship to other systems.
2. If the product is a component of an integrated HW-SW product, then the SRS shall:

(a) summarize the essential characteristics of this larger product;

(b) list the other HW or SW component the software interfaces with, and summarize the computer hardware and peripheral equipment to be used.

3. A block diagram may be presented showing the major components of the larger system or project, interconnections, and external interfaces.

d. Constraints

1. The SRS shall describe any items that limit the developer's options for building the software.
2. The SRS should provide background information and seek to justify the constraints.

e. Logical model description

1. The SRS shall include a top-down description of the logical model of the software.
2. Diagrams, tables and explanatory text may be included.
3. The functionality at each level should be described, to enable the reader to 'walkthrough' e.g. the model level-by-level, function-by-function, and flow-by-flow.
4. Supplementary commentary shall be done to the description of the software product in terms of e.g. data flow diagrams and low-level functional specifications.
5. The behavioural view of the software logical model shall be also described in the SRS.

NOTE This is particularly relevant for flight software applications.

<5> Requirements

a. General

The following provisions apply to the software requirements listed in the SRS, as specified in b. to r. below:

1. Each requirement shall be uniquely identified.
2. The traceability information of each requirement derived from higher level documentation, to the applicable higher level requirement, shall be stated.

b. Functional requirements

1. The SRS shall:

(a) describe the capabilities to be provided by the software item under definition;

(b) include the correlation of the specified software functionalities to system states and modes as applicable.

2. For each functional requirement, its intent shall be reflected in its title.
3. The SRS shall state the purpose of each functional requirements.
4. Functional requirement shall be grouped by subject, in accordance with the logical model organization (e.g. per controlled subsystem).
5. Each requirement definition should be organized according to the following:

(a) General
(b) Inputs
(c) Outputs
(d) Processing

6. The SRS shall describe the functional requirements related to software safety and dependability.

c. Performance requirements

The SRS shall list any specific requirement to the specified performance of software item under definition.

d. Interface requirements

1. The SRS shall list and describe the software item external interfaces.
2. The following interfaces shall be fully described either in the SRS itself or by reference to another document (e.g. ICD):
   (a) interfaces between the software item and other software items;
   (b) interfaces between the software item and hardware products;
   (c) interfaces requirements relating to the man-machine interaction.
3. Naming convention applicable to the data-command interface shall be also described.

e. Operational requirements

1. The SRS shall list any specific requirement related to the operation of the software in its intended environment.
2. The information specified in 1. should include, at least, any specified operational mode and mode transition for the software, and, in case of man-machine interaction, the intended use scenarios.
3. Diagrams may be used to show the intended operations and related modes-transitions.

f. Resources requirements

The SRS shall describe all the resource requirements related to the software and the hardware requirements (target hardware on which the software is specified to operate), as follows:

1. Computer hardware requirements

   List of the requirements relevant to hardware environment in which the software is specified to operate.

2. Computer hardware resource utilization requirements

   List of the sizing and timing requirements applicable to the software item under specification.

3. Computer software requirements

   Description of the computer software to be used with the software under specification or incorporated into the software item (e.g. operating system and software items to be reused).

4. Schedulability requirements

   Description of the real time constraints to respect (e.g. time management with respect to the handling of input data before its loss of validity).

g. Design requirements and implementation constraints

1. The SRS shall list any requirements driving the design of the software item under specification and any identified implementation constraint.
2. Requirements applicable to the following items shall be included:
   (a) software standards (e.g. applicable coding standards, and development standards);
   (b) design requirements;
   (c) specific design methods to be applied to minimize the number of critical software components;
   (d) requirements relevant to numerical accuracy management;
   (e) design requirements relevant to the "in-flight modification" of the software item;
   (f) specific design requirements to be applied if the software is specified to be designed for intended reuse;
   (g) specific requirements on reused software (e.g. COTS, free software and open source).

h. Security and privacy requirements

The SRS shall describe any security and privacy requirement applicable to the software item.

i. Portability requirements

The SRS shall list any portability requirement applicable to the software item.

j. Software quality requirements

The SRS shall list any quality requirement applicable to the software item.

k. Software reliability requirements

The SRS shall list any reliability requirement applicable to the software item.

l. Software maintainability requirements

The SRS shall list any maintainability requirement applicable to the software item.

m. Software safety requirements

The SRS shall list any safety requirement applicable to the software item.

n. Software configuration and delivery requirements

The SRS shall list any requirement applicable to the selected delivery medium and any software configuration applicable to the software item.

o. Data definition and database requirements

The SRS shall list any requirement related to specific data format or structure to be exchanged with other systems or any database requirements allowing to take into account e.g. for a flight software, the mission and product specific constraints.

p. Human factors related requirements

The SRS shall list any requirement applicable to:

1. the personnel and to the specific software product under definition;

2. manual operations, human-equipment interactions, constraints on personnel, concentrated human attention areas and that are sensitive to human errors and training, and human factors engineering.

q. Adaptation and installation requirements

This SRS shall list any requirement applicable to adaptation data and to specific installation.

r. Other requirements

The SRS shall list any additional requirement not covered in b to q above.

<6> Validation approach and requirements

a. General

The SRS shall describe the validation methods and the requirements specified for them in b and c below.

b. Validation approach

The SRS shall summarize the validation approach to be utilized to validate all the requirements stated in <5> and to ensure the requirements are met.

c. Validation requirements

1. The SRS shall describe, per each uniquely identified requirement in <5>, the validation approach.
2. A validation matrix (requirements to validation approach correlation table) shall be utilized to describe the validation approach applicable to each requirement.
3. The validation requirements related to the validation test platform to be used shall be listed (for example, benches or simulators capabilities and their representativity with respect to e.g. real time constraints, target or real hardware equipments on which the software is specified to operate).

<7> Traceability

a. The SRS shall report the traceability matrices

1. from the upper level specification requirements to the requirements contained in <5> (forward traceability table), and
2. from the requirements contained in <5> to the upper level applicable specification (backward traceability table).

b. In case the information in a. is separately provided in the DJF, reference to this documentation shall be clearly stated.

Annex C (normative)

Software design document (SDD) DRD

C.1 DRD identification

C.1.1 Requirement identification and source document

The software design document (SDD) is called from the normative provisions summarized in Table C-1.

NOTE Apart from potential tailoring of ECSS-E-40 Part 1 and ECSS-Q-80 to be performed according to project specificity, these standards specify that the software design document is provided at the following milestones: PDR, CDR and DDR for flight software.

C.1.2 Purpose and objective

The software design document provides description of the software architectural design and the software detailed design. Internal interfaces design is also included in this document.

This software design document is a constituent of the design definition file (DDF).

C.2 Expected response

C.2.1 Response identification

The requirements for project identification contained in ECSS-M-50 shall be applied to the SDD.

C.2.2 Scope and content

The SDD shall provide the information presented in the following sections:

<1> **Introduction**

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

<2> **Applicable and reference documents**

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

Table C-1: SDD traceability to ECSS-E-40 Part 1 and ECSS-Q-80 subclauses

ECSS Standard	Subclauses
ECSS-E-40 Part 1B	5.3.2.7 expected output b
	5.3.2.8
	5.4.2.2 expected output b
	5.4.3.1
	5.4.3.2
	5.4.3.3
	5.4.3.4 expected output a
	5.4.3.4 expected output b
	5.4.3.4 expected output c
	5.4.3.6a
	5.4.3.7
	5.4.3.8 expected output b
	5.5.2.1a
	5.5.2.1b
	5.5.2.1c
	5.5.2.2 expected output b
	5.5.2.3 expected output a
	5.5.2.3 expected output b
	5.5.2.3 expected output c
	5.5.3.1 expected output a
	5.5.3.1 expected output b
	5.5.3.2 expected output a
	5.5.2.5a
	5.5.2.6
	5.5.2.7
	5.8.3.2 expected output a
	5.8.3.3 expected output a
	5.8.3.4 expected output a
ECSS-Q-80B	6.2.3.5 expected output a
	6.2.3.9 expected output a
	7.2.2.1
	7.2.3.2
	7.2.3.3

<3> Terms, definitions and abbreviated terms

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

<4> Software design overview

The SDD briefly introduces the system context and design and discuss the background to the project detailed as follows:

a. Software static architecture

1. The SDD shall describe the architecture of the software item, as well as the main relationship with the major components identified.
2. The SDD shall also describe any system state or mode in which the software operates.

b. Software dynamic architecture

The SDD shall describe the design choices to cope with the real time constraints (e.g. selection and description of the computational model).

c. Interfaces context

1. The SDD shall describe all the external interfaces.
2. The description in 1. should be based on system block diagram or context diagram to illustrate the relationship between this system and other systems.

d. Memory and CPU budget

The SDD shall document and summarize the allocation of memory and processing time to the software components.

e. Design standards, conventions and procedures

1. The SDD shall summarize the software methods adopted for the architectural and the detailed design.

NOTE Being this information provided in the software project development plan, simply a reference to this project document can be given.

2. The following information shall be summarized:
   (a) software architectural design method;
   (b) software detailed design method;
   (c) code documentation standards;
   (d) naming conventions;
   (e) programming standards.

<5> Software design

a. General

1. The SDD shall describe the software architectural design.
2. The architecture structure of the software item shall be described, identifying the software components, their hierarchical relationships, any dependency and interfaces between them.
3. The structure in b. to e. should be used.

b. Overall architecture

1. The SDD shall describe the software architectural design, from a static point of view and also, when the software to be developed has real time constraints, from a dynamic point of view.
2. The software static architecture shall be summarized describing its components.
3. For real-time software, the software dynamic architecture shall be summarized describing its selected computational model.
4. The description in 3. should consist in the following information:
   (a) type of components participating to the real time behaviour,
   (b) scheduling type (e.g. single or multi-threads),

(c) scheduling model (e.g. pre-emptive or not, fixed or dynamic priority based),

(d) analytical model (e.g. rate monotonic scheduling, deadline monotonic scheduling),

(e) Tasks identification and priorities,

(f) Means of communication and synchronization,

(g) Time management.

5. The software static and dynamic architecture shall be described in accordance with the selected design method.

c. Software components design - General

1. The SDD shall describe:

(a) The software components, constituting the software item.

(b) The relationship between the software components.

(c) The purpose of each software component.

(d) For each software component, the development type (e.g. new development, software to be reused).

(e) If the software is written for the reuse,

* its provided functionality from an external point of view, and
* its external interfaces.

NOTE For software intended to be reused, the more "self contained" is the information within this document, the easier to be "extracted" later on.

(f) Handling of existing reused components.

NOTE See ECSS-E-40 Part 2B Annex I.

2. The following apply to the software components specified in 1.(a):

(a) Each software component shall be uniquely identified.

(b) The software requirements allocation shall be provided for each software component;

3. The description of the components should be laid out hierarchically, in accordance with the following aspects for each component:

— <Component identifier>
— <Type>
— <Purpose>
— <Function>
— <Subordinates>
— <Dependencies>
— <Interfaces>
— <Resources>
— <References>
— <Processing>
— <Data>

d. Software components design - Aspects of each component

The following apply to the components aspects identified in c.3.:

1. **<Component identifier>**

(a) Each component should have a unique identifier.

(b) The component should be named according to the rules of the programming language or operating system to be used.

(c) A hierarchical naming scheme should be used that identifies the parent of the component (e.g. ParentName_ChildName).

2. **<Type>**

(a) Component type should be described by stating its logical and physical characteristics.

(b) The logical characteristics should be described by stating the package, library or class that the component belongs to.

(c) The physical characteristics should be described by stating the type of component, using the implementation terminology (e.g. task, subroutine, subprogram, package and file).

NOTE The contents of some components description subclauses depend on the component type. For the purpose of this guide, the following categories are used: executable (i.e. contains computer instructions) or non-executable (i.e. contains only data).

3. **<Purpose>**

The purpose of a component should describe its trace to the software requirements that it implements.

NOTE Backward traceability depends upon each component description explicitly referencing the requirements that justify its existence.

4. **<Function>**

(a) The function of a component shall be described in the software architectural design.

(b) The description specified in (a) should be done by stating what the component does.

NOTE 1 The function description depends upon the component type. Therefore, it can be a description of the process.

NOTE 2 Process descriptions can use such techniques as structured English, precondition-postcondition specifications and state-transition diagrams.

5. **<Subordinates>**

The subordinates of a component should be described by listing the immediate children.

NOTE 1 The subordinates of a module are the modules that are 'called by' it. The subordinates of a database can be the files that 'compose' it.

NOTE 2 The subordinates of an object are the objects that are 'used by' it.

6. **<Dependencies>**

The dependencies of a component should be described by listing the constraints upon its use by other components.

NOTE For example:

- Operations to take place before this component is called.'
- Operations that are excluded when this operation takes place.

7. **<Interfaces>**
   (a) Both control flow and data flow aspects of an interface shall be described for each “executable” component.
   (b) Data aspects of ’non executable’ components should be described.
   (c) The control flow and from a component should be described in terms of how to start (e.g. subroutine call) and terminate (e.g. return) the execution of the component.
   (d) If the information in (c) is implicit in the definition of the type of component, a description need not be done.
   (e) If control flows take place during execution (e.g. interrupt), they should be described.
   (f) The data flow input to and output from each component shall be described.
   (g) It should be ensured that data structures:
      (1) are associated with the control flow (e.g. call argument list);
      (2) interface components through common data areas and files.
8. **<Resources>**
   The resources’ needs of a component should be described by itemising what the component needs from its environment to perform its function.
   NOTE 1 Items that are part of the component interface are excluded.
   NOTE 2 Examples of resources’ needs of a component are displays, printers and buffers.
9. **<References>**
   Explicit references should be inserted where a component description uses or implies material from another document.
10. **<Data>**
   (a) The data internal to a component should be described.
   NOTE The amount of details to be provided depends strongly on the type of the component.
   (b) The data structures internal to a program or subroutine should also be described.
   (c) Data structure definitions shall include the:
      (1) description of each element (e.g. name, type, dimension);
      (2) relationships between the elements (i.e. the structure);
      (3) range of possible values of each element;
      (4) initial values of each element.

e. Internal interface design
   1. The SDD shall describe the internal interfaces among the identified software components.
   2. The interface data specified in 1., by component, shall be organized showing the complete interfaces map, using as appropriate diagrams or matrices supporting their cross-checking.

3. For each identified internal interface, all the defined data elements shall be included.

   NOTE The amount of detail to be provided depends strongly on the type of component.

4. The logical and physical data structure of files that interface major component should be postponed to the detailed design.

5. Data structure definitions shall include:

   (a) the description of each element (e.g. name, type, dimension);

   (b) the relationships between the elements (i.e. the structure);

   (c) the initial values of each element.

<6> Requirements to design components traceability

a. The SDD shall provide traceability matrices

   1. from the software requirements to component down to the lower identified component in the software hierarchy (forward traceability) and

   2. from the software components to its upper level component up to the software requirements (backward traceability).

b. In case the information in a. is provided as separate documentation in the DJF, a reference to it shall be stated.

(This page is intentionally left blank)

Annex D (normative)

Software release document (SRelD) DRD

D.1 DRD identification

D.1.1 Requirement identification and source document

The software release document (SRelD) is called from the normative provisions summarized in Table D-1.

NOTE Apart from potential tailoring of ECSS-E-40 Part 1 and ECSS-Q-80 to be performed according to project specificity, these standards specify that the software release document is provided at the following milestones: QR and AR.

Table D-1: SRelD traceability to ECSS-E-40 Part 1 and ECSS-Q-80 subclauses

ECSS Standard	Subclauses
ECSS-E-40 Part 1B	5.6.4.5 expected output a
	5.7.2.1 expected output b
	5.7.3.6b expected output c

D.1.2 Purpose and objective

The SRelD is a constituent of the DDF.

The purpose of the SRelD is to describe a given software version in terms of known problems, limitations or restrictions with respect to its approved baseline.

D.2 Expected response

D.2.1 Response identification

The requirements for project identification contained in ECSS-M-50 shall be applied to the SRelD.

D.2.2 Scope and content

The SRelD shall provide the information presented in the following sections:

<1> Introduction

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

<2> Applicable and reference documents

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

<3> Terms, definitions and abbreviated terms

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

<4> Software release overview

The SRelD shall contain a brief description of the information to be associated with a software release, including:

a. reference of the corresponding SCF,

b. version of the delivered software configuration item,

c. status of SPRs, SCRs and SW&D related to the software configuration item, and

d. advice for use of the software configuration item.

NOTE The software release document is a subset of the software configuration file that describes a new version by comparison with "reference" or the previous one. It is used for the delivery of a new version of a software configuration item to a customer.

<5> Status of the software configuration item

a. Evolution since previous version

The SRelD shall

1. summarize the main information on the software configuration item, and
2. describe the changes implemented since previous version.

b. Known problems or limitations

The SRelD shall list all the unsolved SPR and approved SW&D related to the version of the software configuration item.

<6> Advice for use of the software configuration item

The SRelD shall provide advice for the use of this version of the software configuration item (e.g. potential problems, and compatibility with other configuration items).

<7> On-going changes

The SRelD shall provide information on planned evolution of the software configuration item.

Annex E (normative)

Software [unit/integration] test plan (SUITP) DRD

E.1 DRD identification

E.1.1 Requirement identification and source document

The software [unit/integration] test plan (SUITP) is called from the normative provisions summarized in Table E-1.

Table E-1: SUITP traceability to ECSS-E-40 Part 1 and ECSS-Q-80 subclauses

ECSS Standard	Subclauses
ECSS-E-40 Part 1B	5.4.3.13
	5.5.2.9
	5.5.2.10
	5.5.3.1 expected output c
	5.5.3.4
	5.5.4.1
ECSS-Q-80B	6.3.4.19
	6.3.4.21
	7.3.1.1
	7.3.1.2
	7.3.1.3
	7.3.1.4
	7.3.1.5
	6.3.4

NOTE Apart from potential tailoring of ECSS-E-40 Part 1 and ECSS-Q-80 to be performed according to project specificity, these standards specify that the software unit test plan is provided at the DDR and CDR milestones, and the software integration test plan is provided at the PDR, DDR for flight software and CDR milestones.

E.1.2 Purpose and objective

The software unit test plan and the software integration test plan are constituents of the design justification file.

The purpose of this DRD is to describe the tests plans, and is utilized for the following documents:

- the software unit test plan;
- the software integration test plan.

NOTE Although this DRD provides a unique template for unit and integration testing, to be instantiated with respect to the software test plans specified in the document requirement list, either for a software unit test plan, either for a software integration test plan. The acronym SUITP is used to designate either the software unit test plan, either the software integration test plan.

E.2 Expected response

E.2.1 Response identification

The requirements for project identification contained in ECSS-M-50 shall be applied to the SUITP.

E.2.2 Scope and content

The SUITP shall provide the information presented in the following sections:

<1> Introduction

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

<2> Applicable and reference documents

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

<3> Terms, definitions and abbreviated terms

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

<4> Software overview

The SUITP shall contain a brief description of the software under test and its context: a summary of its functionality, its configuration, its operational environment and its external interfaces.

NOTE Reference to technical documentation can be done.

<5> Software unit testing and software integration testing

The SUITP describes the responsibility and schedule information for the software unit testing and integration testing, detailed as follows:

a. Organization
   1. The SUITP shall describe the organization of software unit testing and integration testing activities.
   2. The following topics should be included:
      (a) roles,
      (b) reporting channels,
      (c) levels of authority for resolving problems,

(d) relationships to the other activities such as project management, development, configuration management and product assurance.

b. Master schedule

1. The SUITP shall describe the schedule for the software unit testing and integration testing activities, in particular, test milestones identified in the software project schedule and all item delivery events.
2. The SUITP should include:
   (a) a reference to the master schedule given in the software development plan,
   (b) any additional test milestones and state the time required for each testing task,
   (c) the schedule for each testing task and test milestone,
   (d) the period of use for the test facilities.

c. Resource summary

The SUITP shall summarize the resources needed to perform the software unit testing / integration testing activities such as staff, hardware and software tools.

d. Responsibilities

1. The SUITP shall describe the specific responsibilities associated with the roles described in a.
2. The responsibilities specified in 1. should be described by identifying the groups responsible for managing, designing, preparing, executing the tests.

NOTE Groups can include developers, technical support staff, and product assurance staff.

e. Tools, techniques and methods

The SUITP shall describe the hardware platforms, software tools, techniques and methods used for software unit testing and integration testing activities.

f. Personnel and personnel training requirements

The SUITP shall list any requirement for software unit testing and integration testing personnel and their training needs.

g. Risks and contingencies

1. The SUITP shall describe risks to the software unit testing and integration testing campaign.
2. Contingency plans should be included.

<6> Control procedures for software unit testing / integration testing

The SUITP shall contain information (or reference to) about applicable management procedures concerning the following aspects:

a. problem reporting and resolution;

b. deviation and waiver policy;

c. control procedures.

<7> Software unit testing and integration testing approach

The SUITP describes the approach to be utilized for the software unit testing and integration testing, detailed as follows:

a. Tasks and items under test

The SUITP shall describe which are the tasks and the items under tests, as well as criteria to be utilized.

b. Features to be tested

The SUITP shall describe all the features to be tested, making references to the applicable documentation.

c. Features not to be tested

The SUITP shall describe all the features and significant combinations not to be tested.

d. Test pass - fail criteria

The SUITP shall describe the general criteria to be used to determine whether or not test are passed.

e. Suspension criteria and resumption requirements

1. The SUITP shall describe the criteria used to suspend all, or a part of, the testing activities on the test items associated with the plan.
2. The SUITP should describe the testing activities to be repeated when testing is resumed.

<8> Software unit test / integration test design

a. General

1. The SUITP shall provide the definition of unit and integration test design.
2. For each identified test design, the SUITP shall provide the information given in b.

NOTE This can be simplified in the software unit test plan.

b. Organization of each identified test design

The SUITP provides the definition of each unit test and integration test design, detailed as follows:

1. Test design identifier
   (a) The SUITP shall identify each test design uniquely.
   (b) The SUITP shall briefly describe the test design.
2. Features to be tested
   (a) The SUITP shall list the test items and describe the features to be tested.
   (b) Reference to appropriate documentation shall be made and traceability information shall be provided.
3. Approach refinements
   (a) The SUITP shall describe the test approach implemented for the specific test design and the specific test class.
   (b) The description specified in (a) shall provide the rationale for the test case selection and grouping into test procedures.
   (c) The method for analysing test results shall be identified (e.g. compare with expected output, or compare with old results).
   (d) Configuration of the facility (both hardware and software) to be used to execute the identified test shall be described.
4. Test case identifier

   The SUITP shall list the test cases associated with the test design and provide a summary description of each ones.

<9> Software unit and integration test case specification

a. General

1. The SUITP shall provide an identification of software unit test and integration test cases.
2. For each identified test case, the SUITP shall provide the information given in b.

NOTE Each test case can be described through one or several description sheets.

b. Organization of each identified test case

The SUITP provides the definition of each unit and integration test case, detailed as follows:

1. Test case identifier
   (a) The SUITP shall identify the test case uniquely.
   (b) A short description of the test case purpose shall be provided.
2. Test items
   (a) The SUITP shall list the test items.
   (b) Reference to appropriate documentation shall be performed and traceability information shall be provided.
3. Inputs specification

   The SUITP shall describe the inputs to execute the test case.
4. Outputs specification

   This SUITP shall describe the expected outputs.
5. Test pass – fail criteria

   The SUITP shall list the criteria to decide whether the test has passed or failed.
6. Environmental needs

   The SUITP shall describe:
   (a) the exact configuration and the set up of the facility used to execute the test case as well as the utilization of any special test equipment (e.g. bus analyser);
   (b) the configuration of the software utilized to support the test conduction (e.g. identification of the simulation configuration);
   (c) any other special requirement.
7. Special procedural requirements

   The SUITP shall describe any special constraints on the used test procedures.
8. Interfaces dependencies

   The SUITP shall describe all the test cases to be executed before this test case.
9. Test script

   The SUITP shall descibe all the test script used to execute the test case.

   NOTE The test scripts can be collected in an appendix.

<10> Software unit and integration test procedures

a. General

1. The SUITP shall provide a identification of software unit and integration test procedures.
2. For each identified test procedure, the SUITP shall provide the information given in b.

b. Organization of each identified test procedure

The SUITP provides the definition of each unit and integration test procedure, detailed as follows:

1. Test procedures identifier

   The SUITP shall include a statement specifying the test procedure uniquely.

2. Purpose

   (a) The SUITP shall describe the purpose of this procedure.

   (b) A reference to each test case implemented by the test procedure shall be given.

3. Special requirements

   The SUITP shall include any special requirement for the execution of this procedure.

4. Procedure steps

   The SUITP shall describe every step of the procedure execution:

   (a) log: describe any special methods or format for logging the results of test execution, the incidents observed, and any other event pertinent to this test;

   (b) set up: describe the sequence of actions to set up the procedure execution;

   (c) start: describe the actions to begin the procedure execution;

   (d) proceed: describe the actions during the procedure execution;

   (e) measure: describe how the test measurements is made;

   (f) shut down: describe the action to suspend testing when interruption is forced by unscheduled events;

   (g) restart: identify any procedural restart points and describe the actions to restart the procedure at each of these points;

   (h) wrap up: describe the actions to terminate testing;

   (i) contingencies: describe the actions to deal with anomalous events that may occur during execution.

<11> Software test plan additional information

The following additional information shall be provided:

a. test procedures to test cases traceability matrix;

b. test cases to test procedures traceability matrix;

c. test scripts;

d. detailed test procedures.

NOTE 1 This information can be given in separate appendices.

NOTE 2 One test design uses one or more test cases.

NOTE 3 One test procedure execute one or more test cases.

Annex F (normative)

Software validation testing specification (SVTS) DRD

F.1 DRD identification

F.1.1 Requirement identification and source document

The software validation testing specification (SVTS) is called from the normative provisions summarized in Table F-1.

Table F-1: SVTS traceability to ECSS-E-40 Part 1 and ECSS-Q-80 subclauses

ECSS Standard	Subclauses
ECSS-E-40 Part 1B	5.8.3.7 expected output a
	5.8.3.7 expected output b
	5.6.3.1
	5.6.3.2a
	5.6.4.1
ECSS-Q-80B	6.3.4
	6.3.4.17
	6.3.4.18
	6.3.4.19
	7.3.1.1
	7.3.1.5

NOTE Apart from potential tailoring of ECSS-E-40 and ECSS-Q-80 to be performed according to project specificity, these standards specify that the software validation testing specification with respect to the technical specification is provided at the CDR milestone, and the software validation testing specification with respect to the requirement baseline is provided at the QR milestone.

F.1.2 Purpose and objective

The software validation testing specification with respect to the technical specification and the software validation testing specification with respect to the requirement baseline are constituents of the design justification file.

The purpose of this DRD is to describe the testing specifications and is utilized to document

- the software validation testing specification with respect to the technical specification (TS), and
- the software validation testing specification with respect to the requirements baseline (RB).

NOTE Although this DRD provides a unique template for the software validation testing specification document, to be instantiated with respect to, either the technical specification, either the requirement baseline. The acronym SVTS w.r.t. TS is used to designate the software validation testing specification with respect to the technical specification whilst SVTS w.r.t. RB is used to designate the software validation testing specification with respect to the requirement baseline.

F.2 Expected response

F.2.1 Response identification

The requirements for project identification contained in ECSS-M-50 shall be applied to the SVTS w.r.t. TS and the SVTS w.r.t. RB.

F.2.2 Scope and content

The SVTS w.r.t. TS or RB shall provide the information presented in the following sections:

<1> Introduction

The SVTS w.r.t. TS or RB shall contain a description of the purpose, objective, content and the reason prompting its preparation.

<2> Applicable and reference documents

The SVTS w.r.t. TS or RB shall list the applicable and reference documents to support the generation of the document.

<3> Terms, definitions and abbreviated terms

The SVTS w.r.t. TS or RB shall include any additional terms, definition or abbreviated terms used.

<4> Software overview

The SVTS w.r.t. TS or RB shall contain a brief description of the software under test and its context: a summary of its functionality, its configuration, its operational environment and its external interfaces.

NOTE Reference to technical documentation can be done.

<5> Software validation testing specification task identification

The SVTS w.r.t. TS or RB describes the approach to be utilized for the software validation testing specification, detailed as follows:

a. Task and criteria

The SVTS w.r.t. TS or RB shall describe which are the tasks and the items under tests, as well as criteria to be utilized.

a. Features to be tested

The SVTS w.r.t. TS or RB shall describe all the features to be tested, making references to the applicable documentation.

b. Features not to be tested

The SVTS w.r.t. TS or RB shall describe all the features and significant combinations not to be tested.

c. Test pass – fail criteria

The SVTS w.r.t. TS or RB shall describe the general criteria to be used to determine whether or not tests are passed.

d. Suspension criteria and resumption requirements

1. The SVTS w.r.t. TS or RB shall describe the criteria used to suspend all, or a part of, the testing activities on the test items associated with the plan.
2. The SVTS w.r.t. TS or RB should describe the testing activities to be repeated when testing is resumed.

<6> Software validation testing specification design

a. General

1. The SVTS w.r.t. TS or RB shall provide the definition of software validation testing specification design.
2. For each identified test design, the SVTS w.r.t. TS or RB shall provide the information given in b.

b. Organization of each identified test design

The SVTS w.r.t. TS or RB provides the definition of each unit test and integration test design, detailed as follows:

1. Test design identifier
   (a) The SVTS w.r.t. TS or RB shall describe the test design uniquely.
   (b) The SVTS w.r.t. TS or RB shall briefly describe the test design.
2. Features to be tested
   (a) The SVTS w.r.t. TS or RB shall describe the test items and the features to be tested.
   (b) Reference to appropriate documentation shall be performed and traceability information shall be provided.
3. Approach refinements
   (a) The SVTS w.r.t. TS or RB shall describe the test approach implemented for the specific test design and the specific test class implemented.
   (b) The description specified in (a) shall provide the rationale for the test case selection and grouping into test procedures.

(c) The method for analysing test results shall be identified (e.g. compare with expected output, and compare with old results).

(d) Configuration of the facility (both hardware and software) to be used to execute the identified test shall be described.

4. Test case identification

The SVTS w.r.t. TS or RB shall

(a) list the test cases associated with the test design, and

(b) provide a summary description of each one.

<7> Software validation test case specification

a. General

1. The SVTS w.r.t. TS or RB shall provide the identification of software validation test cases.
2. For each identified test case, the SVTS w.r.t. TS or RB shall provide the information given in b.

b. Organization of each identified test case

The SVTS w.r.t. TS or RB provides the definition of each validation test case, detailed as follows:

1. Test case identifier

(a) The SVTS w.r.t. TS or RB shall describe the test case uniquely.

(b) A short description of the test case purpose shall be provided.

2. Test items

(a) The SVTS w.r.t. TS or RB shall describe, for each test case, the test items.

(b) Reference to appropriate documentation shall be performed and traceability information shall be provided.

3. Inputs specification

The SVTS w.r.t. TS or RB shall describe, for each test case, the inputs to execute the test case.

4. Outputs specification

The SVTS w.r.t. TS or RB shall describe, for each test case, the expected outputs.

5. Test pass - fail criteria

The SVTS w.r.t. TS or RB shall describe, for each test case, the criteria to decide whether the test has passed or failed.

6. Environmental needs

The SVTS w.r.t. TS or RB shall describe:

(a) the exact configuration and the set up of the facility used to execute the test case as well as the utilization of any special test equipment (e.g. bus analyser);

(b) the configuration of the software utilized to support the test conduction (e.g. identification of the simulation configuration);

(c) any other special requirement.

7. Special procedural requirements

   The SVTS w.r.t. TS or RB shall describe any special constraints on the used test procedures.

8. Interfaces dependencies

   The SVTS w.r.t. TS or RB shall list all the test cases to be executed before this test case.

9. Test script

   The SVTS w.r.t. TS or RB shall list all the test script used to execute the test case.

   NOTE The test scripts can be collected in an appendix.

<8> Software validation test procedures

a. General

   1. The SVTS w.r.t. TS or RB shall provide the identification of software validation test procedures.
   2. For each identified validation test procedure, the SVTS w.r.t. TS or RB shall provide the information presented in b.

b. Organization of each identified test procedure

   The SVTS w.r.t. TS or RB provides the description of each identified validation test procedure, detailed as follows:

   1. Test procedure identifier

      The SVTS w.r.t. TS or RB shall identify each test procedure uniquely.

   2. Purpose

      (a) The SVTS w.r.t. TS or RB shall describe the purpose of each test procedure.

      (b) A reference to each test case used by the test procedure shall be given.

   3. Special requirements

      The SVTS w.r.t. TS or RB shall state any special requirement for the execution of each test procedure.

   4. Procedure steps

   The SVTS w.r.t. TS or RB shall describe every step of each procedure execution:

      (a) log: describe any special methods or format for logging the results of test execution, the incidents observed, and any other event pertinent to this test;

      (b) set up: describe the sequence of actions necessary to set up the procedure execution;

      (c) start: describe the actions necessary to begin the procedure execution;

      (d) proceed: describe the actions necessary during the procedure execution;

      (e) measure: describe how the test measurements is made;

      (f) shut down: describe the action necessary to suspend testing when interruption is forced by unscheduled events;

      (g) restart: identify any procedural restart points and describe the actions necessary to restart the procedure at each of these points;

(h) wrap up: describe the actions necessary to terminate testing;

(i) contingencies: describe the actions necessary to deal with anomalous events that may occur during execution.

<9> Software validation testing specification additional information

The following additional information shall be included in the SVTS w.r.t. TS or RB:

a. Test to requirement traceability matrix,

b. Requirement to test traceability matrix,

c. Test procedures to test cases traceability matrix,

d. Test cases to test procedures traceability matrix,

e. Test scripts,

f. Detailed test procedures.

NOTE 1 This information can be given in separate appendices.

NOTE 2 One test design uses one or more test cases.

NOTE 3 One test procedure execute one or more test cases.

Annex G (normative)

Software verification plan (SVerP) DRD

G.1 DRD identification

G.1.1 Requirement identification and source document

The software verification plan (SVerP) is called from the normative provisions summarized in Table G-1.

NOTE Apart from potential tailoring of ECSS-E-40 Part 1 and ECSS-Q-80 to be performed according to project specificity, these standards specify that the software verification plan is provided at the PDR.

Table G-1: SVerP traceability to ECSS-E-40 Part 1 and ECSS-Q-80 subclauses

ECSS Standard	Subclauses
ECSS-E-40 Part 1B	5.8.2.1a
	5.8.2.1b
	5.8.2.1c
	5.8.2.2a
	5.8.2.2b
	5.8.2.2c
	5.8.2.3a
	5.8.2.3b
	5.8.2.4
ECSS-Q-80B	6.2.1.1
	6.2.3.2
	6.2.3.9
	6.2.6.1
	6.3.3.23

G.1.2 Purpose and objective

The software verification plan is a constituent of the design justification file (DJF).

The purpose of the software verification plan is to describe the approach and the organization aspects to implement the software verification activities.

G.2 Expected response

G.2.1 Response identification

The requirements for project identification contained in ECSS-M-50 shall be applied to the SVerP.

G.2.2 Scope and content

The SVerP shall provide the information presented in the following sections:

<1> **Introduction**

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

<2> **Applicable and reference documents**

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

<3> **Terms, definitions and abbreviated terms**

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

<4> **Software verification process overview**

The SVerP shall describe the approach to be utilized to implement the verification process throughout the software life cycle, the verification effort, and the level of independence for the verification tasks, as follows:

a. Organization
   1. The SverP shall describe the organization of the documentation review, proofs, and tracing activities.
   2. The following topics that shall be included:
      (a) roles;
      (b) reporting channels;
      (c) levels of authority for resolving problems;
      (d) organization relationships;
      (e) level of required and implemented independence.

b. Master schedule
   1. A reference to the master schedule given in the software development plan shall be done.
   2. This SVerP shall describe the schedule for the planned verification activities.
   3. Information about reviews shall be included.

c. Resource summary

   The SVerP shall summarize the resources to be used to perform the verification activities such as staff, hardware and software tools.

d. Responsibilities

   The SVerP shall describe the specific responsibilities.

e. Tools, techniques and methods

The SVerP shall describe the software tools, techniques and methods used to execute the verification tasks throughout the software life cycle.

<5> Control procedures for verification process

The SVerP shall contain information (or reference to) about applicable management procedures concerning the following aspects:

a. problem reporting and resolution;

b. deviation and waiver policy;

c. control procedures.

<6> Verification activities

a. General

1. The SVerP shall include per each software item the information in b. to f. below.
2. Each of the items specified in 1 above should include the following:
   (a) For each of the tasks pertaining to the verification process, the description of the methods, tools and facilities utilized, and the outputs to be produced.

   NOTE This includes, for example, the identification of internal reviews, walkthrough, and inspections.

   (b) When the need of utilization of specific methods and tools is identified (e.g. formal proof) an entry with such a statement.

b. Software requirements and architecture engineering process verification

1. Activities

   The SVerP shall list the verification activities to be performed for this software process and how these are accomplished.

2. Inputs

   The SVerP shall list the required inputs (e.g. draft SRS, draft software architectural design) to accomplish the verification activities for this software process.

3. Outputs

   The SVerP shall list the intermediate and final outputs (e.g. software verification requirements report, software architectural design to requirements traceability) documenting the performed verification activities for this software process.

4. Methodology, tools and facilities

   The SVerP shall describe the methodologies, tools and facilities utilized to accomplish the verification activities.

c. Software design and implementation engineering process verification

1. Activities

   The SVerP shall list the verification activities to be performed for this software process and how these are accomplished.

2. Inputs

   The SVerP shall list the required inputs (e.g. software components design, code, software user manual, software integration test plan) to accomplish the verification activities for this software process.

3. Outputs

   The SVerP shall list the intermediate and final outputs (e.g. software code verification report, evaluation of software validation testing specification) documenting the performed verification activities for this software process.

4. Methodology, tools and facilities

   The SVerP shall describe the methodologies, tools and facilities utilized to accomplish the verification activities.

d. Software delivery and acceptance process verification

1. Activities

   The SVerP shall list the verification activities to be performed for this software process and how these are accomplished.

2. Inputs

   The SVerP shall list the required inputs (e.g. software validation specification with respect to the requirements baseline, software acceptance testing documentation,) to accomplish the verification activities for this software process.

3. Outputs

   The SVerP shall list the intermediate and final outputs (e.g. software acceptance test report, software acceptance data package, problem reports, software release document, software configuration file, etc) documenting the performed verification activities for this software process.

4. Methodology, tools and facilities

   The SVerP shall describe the methodologies, tools and facilities utilized to accomplish the verification activities.

e. Software validation process verification

1. Activities

   The SVerP shall list the verification activities to be performed for this software validation process and how these are accomplished.

2. Inputs

   The SVerP shall list the required inputs (e.g. software validation specification with respect to the requirements baseline) to accomplish the verification activities for this software process.

3. Outputs

   The SVerP shall list the intermediate and final outputs (e.g. software validation testing specifications) documenting the performed verification activities for this software process.

4. Methodology, tools and facilities

   The SVerP shall describe the methodologies, tools and facilities utilized to accomplish the verification activities.

f. Software product assurance programme implementation verification

1. Activities

   The SVerP shall list the verification activities to be performed for this software process (e.g. review of the traceability matrices) and how these are accomplished.

2. Inputs

   The SVerP shall list the required inputs (e.g. tests results to be checked to verify the required coverage has been met) to accomplish the verification activities for this software process.

3. Outputs

   The SVerP shall list the intermediate and final outputs (e.g. contribution to the software product assurance reports, contribution to metrics) documenting the performed verification activities for this software process.

4. Methodology, tools and facilities

   The SVerP shall describe the methodologies, tools and facilities utilized to accomplish the software product assurance verification activities.

<7> Software verification reporting

a. The SVerP shall describe how the results of implementing should be documented. Reports produced by the implementation of the verification process shall be identified.

b. The reports identified in ECSS-E-40 Part 1 and ECSS-Q-80 shall be referenced, as implemented by the specific software project for which this plan is written as well as any problem report mechanism adopted (e.g. utilization of RID from or problem report form).

c. Forwarding procedure of this documentation to the customer shall be also addressed.

(This page is intentionally left blank)

Annex H (normative)

Software validation plan (SValP) DRD

H.1 DRD identification

H.1.1 Requirement identification and source document

The software validation plan (SValP) is called from the normative provisions summarized in Table H-1.

NOTE Apart from potential tailoring of ECSS-E-40 Part 1 and ECSS-Q-80 to be performed according to project specificity, these standards specify that the software validation plan is provided at the PDR.

Table H-1: SValP traceability to ECSS-E-40 Part 1 and ECSS-Q-80 subclauses

ECSS Standard	Subclauses
ECSS-E-40 Part 1B	5.6.2.1
	5.6.2.2a
	5.6.2.2b
	5.6.2.4
ECSS-Q-80B	6.3.4.23
	6.3.4.25
	7.3.1

H.1.2 Purpose and objective

The software validation plan is a constituent of the design justification file (DJF).

The purpose of the software validation plan is to provide the definition of organizational aspects and management approach to the implementation of the validation tasks.

The objective of the software validation plan is to describe the approach to the implementation of the validation process for a software product.

H.2 Expected response

H.2.1 Response identification

The requirements for project identification contained in ECSS-M-50 shall be applied to the SValP.

H.2.2 Scope and content

The SValP shall provide the information given in the following sections:

<1> Introduction

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

<2> Applicable and reference documents

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

<3> Terms, definitions and abbreviated terms

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

<4> Software validation process planning

a. General

1. The SValP shall describe the approach to be utilized to implement the validation process, the required effort, and the level of required independence for the validation tasks.
2. The SValP shall also address, if it is applicable to the software validation campaign against the requirements baseline, to the software validation campaign against the technical specification, or, for both.

b. Organization

1. The SValP shall describe the organization of the validation activities.
2. Topics that shall be included are:
   (a) organizational structure;
   (b) relationships to the other activities such as project management, development, configuration management and product assurance;
   (c) level of required and implemented independence in validation activities execution.

c. Schedule

1. A reference to the master schedule given in the software development plan shall be included.
2. The SValP shall describe the schedule for the planned validation activities. In particular, test milestones identified in the software project schedule and all item delivery events.
3. The SValP shall describe:
   (a) any additional test milestones and state the time required for each testing task;
   (b) the schedule for each testing task and test milestone;
   (c) the period of use for the test facilities.

d. Resource summary

The SValP shall summarize the resources needed to perform the validation activities such as staff, hardware, software tools, testing data and support software (simulators).

e. Responsibilities

1. The SValP shall describe the specific responsibilities associated with the roles described in b. above.
2. In particular, the SValP shall state the groups responsible for managing, designing, preparing, executing witnessing and checking tests.

NOTE Groups can include developers, operational staff, user representatives, technical support staff and product assurance staff.

f. Tools, techniques and methods

The SValP shall describe the software tools, techniques and methods used for validation activities as well as the needed hardware facilities and, testing data, support software (simulators).

g. Personnel requirements

The SValP shall describe any requirement for software validation personnel (level of independence) and any necessary training needs.

h. Risks

1. The SValP shall state all the identified risks to the software validation campaign.
2. Contingency plans shall be also included.

<5> Software validation tasks identification

a. The SValP shall describe the software validation tasks to be performed for the identified software items.

b. The SValP shall list which are the tasks and the items under tests, as well as the criteria to be utilized for the testing activities on the test items associated with the plan.

c. The SValP shall list the testing activities to be repeated when testing is resumed.

d. The SValP shall describe for each validation tasks the inputs, the outputs as well as the resources to be used for each task.

e. The detailed information and the data for the testing procedures shall be provided in the software validation testing specifications.

<6> Software validation approach

a. The SValP shall describe the overall requirements applicable to the software validation testing activities, providing for definition of overall requirements, guidelines on the kinds of tests to be executed.

b. The SValP shall include the definition of the test cases and test procedures defined in the software validation testing specification for the validation tasks to be executed to verify the technical specification and the requirements baseline.

c. The SValP shall describe the selected approach to accomplish validation of those software specification requirements to be validated by inspection and analysis or review of design or demonstration.

<7> Software validation testing facilities

a. This SValP shall describe the test environment to execute the software validation testing activity and the non-testing validation activities whose approach is defined by this plan.

b. The SValP shall describe the configuration of selected validation facilities in terms of software (e.g. tools and programs, and simulation), hardware (e.g. platforms and target computer), test equipment (e.g. bus analyser), communications networks, testing data and support software (e.g. simulators).

NOTE Reference to other documentation describing the facility can be done.

c. If the validation testing against the requirements baseline and the validation testing against the technical specification use different environments, this shall be clearly stated and described.

<8> Control procedures for software validation process

The SValP shall contain information (or reference to) about applicable management procedures concerning the following aspects:

a. problem reporting and resolution;

b. deviation and waiver policy;

c. configuration control and management.

Annex I (normative)

Software reuse file (SRF) DRD

I.1 DRD identification

I.1.1 Requirement identification and source document

The software reuse file (SRF) is called from the normative provisions summarized in Table I-1.

Table I-1: SRF traceability to ECSS-E-40 Part 1 and ECSS-Q-80 subclauses

ECSS Standard	Subclauses
ECSS-Q-80B	6.2.7.1
	6.2.7.2
	6.2.7.3
	6.2.7.4
	6.2.7.5
	6.2.7.6
	6.2.7.7
	6.2.7.8
	6.2.7.9
	6.2.7.10
	6.2.7.11
	7.2.3.4
	7.2.3.6
	7.2.3.7
ECSS-E-40 Part 1B	5.4.3.9
	5.4.3.10
	5.4.3.12

NOTE Apart from potential tailoring of ECSS-E-40 Part 1 and ECSS-Q-80 to be performed according to project specificity, these standards specify that the software reuse file is provided at the SRR, PDR and CDR milestones.

I.1.2 Purpose and objective

The software reuse file is a constituent of the design justification file (DJF) to be developed in the course of a software development project.

The purpose of the software reuse file is to document the analysis to be performed on existing software intended to be reused.

The global objective of the software reuse file is to document all the information used to decide about the reuse (or not) of existing software and specific action plan related to identified risks.

I.2 Expected response

I.2.1 Response identification

The requirements for project identification contained in ECSS-M-50 shall be applied to the SRF.

I.2.2 Scope and content

The SRF shall provide the information presented in the following sections:

<1> Introduction

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

<2> Applicable and reference documents

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

<3> Terms, definitions and abbreviated terms

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

<4> Presentation of the software intended to be reused

a. The SRF shall describe the administrative and technical information available on the software intended for reuse.

b. For each software item, the SRF shall provide the following information:

1. software item name and main features;
2. developer name;
3. industrial property constraints, if any;
4. applicable dispositions for maintenance, installation and training;
5. commercial software necessary for software execution, if any;
6. development and execution environment (e.g. computer, and operating system);
7. considered version and list of components;
8. implementation language;
9. size of the software (e.g. number of source code lines, and size of the executable code).

<5> Compatibility of existing software item with project objectives

a. General

1. The SRF shall document the result of the analysis performed on existing software regarding project objectives.
2. For each of the software items identified in <4> (e.g. software items a, b, c), the information presented in b. shall be provided.

b. Information for each software item

For each software item, the SRF provides the following information:

1. Functionality and technical constraints
   (a) The SRF shall summarize the results of the technical analysis performed on the existing software to evaluate which part of the requirements baseline (RB) of the project it contributes to cover and to which extent.

   NOTE This information deals also with compatibility or impacts on project technical constraints, such as memory size and calculation time.

   (b) The SRF shall also provide detailed results.

   NOTE Detailed results can be provided in an appendix.
2. Quality and validation
   (a) The SRF shall sum up the results of the quality analysis as well as the criticality level and system criticality performed on the existing software to evaluate the extent to which it fits with project product quality requirements and project verification and validation requirements.
   (b) The following quality features shall be included, as a minimum:
   * the availability of quality technical documentation;
   * adherence to coding standards.
   (c) Detailed results shall be provided.

   NOTE Detailed results can be provided in appendices.

<6> Conclusion: reuse strategy

a. The SRF shall present the project team strategy regarding reuse.

b. The presentation specified in a. shall be done for each considered software by describing:

1. the final decision: reuse or not with main justification based on material exposed in previous chapters;
2. the level of reuse and the level and amount of functional update to be done;
3. the list of actions to be implemented to cope with deviation regarding quality or validation (this can be achieved through referencing a specific action plan if it exists).

<7> Methods and tools used for the software evaluation

a. The SRF shall describe or reference the way the project performed the functional and quality evaluation whose results are described in <5>.

b. Methods and tools used to evaluate the functional and quality characteristics of the considered software shall be described.

c. Role and responsibility of personnel (in the organization and the project) who performed it shall be described.

NOTE All the information on methods and tools used for software evaluation can be presented in an appendix.

<8> Detailed results of evaluation

The SRF shall include the detailed results of the evaluation.

NOTE The detailed results of the evaluation can be presented in an appendix.

Annex J (normative)

Software development plan (SDP) DRD

J.1 DRD identification

J.1.1 Requirement identification and source document

The software development plan (SDP) is called from the normative provisions summarized in Table J-1.

NOTE Apart from potential tailoring of ECSS-E-40 Part 1 and ECSS-Q-80 to be performed according to project specificity, these two standards specify that the software development plan is provided at the following milestones: SRR and PDR.

J.1.2 Purpose and objective

The software development plan is a constituent of the management file (MGT).

The purpose of the software development plan is to describe the established management and development approach for the software items to be defined by a software supplier to set up a software project in accordance with the customer requirements.

J.2 Expected response

J.2.1 Response identification

The requirements for project identification contained in ECSS-M-50 shall be applied to the SDP.

J.2.2 Scope and content

The SDP shall provide the information presented in the following sections:

<1> **Introduction**

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

<2> **Applicable and reference documents**

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

Table J-1: SDP traceability to ECSS-E-40 Part 1 and ECSS-Q-80 subclauses

ECSS Standard	Subclauses
ECSS-E-40B Part 1	5.3.2.1
	5.3.2.2a
	5.3.2.2b
	5.3.2.2c
	5.3.2.3
	5.3.2.4
	5.3.2.5
	5.3.2.11
	5.3.2.14
	5.3.2.15
	5.3.4.2 (interface management procedures)
ECSS-Q-80B	5.2.5.1
	5.4.5.2
	5.5
	5.7.4
	5.8.2
	6.1.4
	6.1.5
	6.2.1.1
	6.2.1.2
	6.2.1.3
	6.2.4.2
	6.3.2.1
	7.2.3.5

<3> Terms, definitions and abbreviated terms

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

<4> Software project management approach

a. Management objectives and priorities

1. The SDP shall describe the management objectives of the software project and associated priorities.
2. The SDP shall include a discussion of the objectives.

b. Master schedule

The SDP shall make a reference to the general project master schedule.

c. Assumptions, dependencies and constraints

The SDP shall state:

1. the assumptions on which the plan is based;
2. the external events the project is dependent upon;

3. constraints on the project;
4. technical issues.

NOTE Technical issues are only mentioned if they have an effect on the plan. Assumptions, dependencies and constraints are often difficult to distinguish. The best approach is not to categorize them but to list them. For example:

* limitations on the budget;
* schedule constraints (e.g. launch dates, delivery dates);
* constraints on the location of staff (e.g. they must work at developer's premises);
* commercial hardware or software used by the system;
* availability of simulators and others test devices;
* availability of external systems with which the system must interface.

d. Work breakdown structure

The SDP shall list the activities to be performed in order to develop the software configuration item.

NOTE 1 See ECSS-M-10 for further explanation.

NOTE 2 Sometimes the adequate elementary tasks can be identified only if several levels of activities breakdown are performed.

e. Risk management

The SDP shall describe the contribution of the software engineering function to the project risk management.

NOTE See ECSS-M-00-03 for further explanations.

f. Monitoring and controlling mechanisms

The SDP shall describe the monitoring mechanisms for managing the work (e.g. progress report, progress meeting, action item lists).

NOTE The SDP apply to both the customer relationships and the supplier's relationships.

g. Staffing plan

The SDP shall describe the roles and skills of staff involved in the project.

h. Software procurement process

The SDP shall describe the software procurement process implementation.

i. Supplier management

The SDP shall describe the supplier management approach.

NOTE The software management aspects specified in a. to h. above can be fully described in the SDP or, at higher level, in a project management plan according to the ECSS-M Standards

<5> Software development approach

a. Strategy to the software development

The SDP shall describe the overall strategy to the software development.

NOTE An activity diagram can be included.

b. Software project development life cycle

1. Software development life cycle identification
   (a) The SDP shall describe the software development life cycle.
   (b) Definition of the selected life cycle paradigm (e.g. waterfall, incremental, or evolutionary) as well as the adopted software versioning approach shall be included.
   (c) The SDP shall cover the implementation of all the activities and tasks relevant to the involved software processes, including:
      * system engineering processes related to software;
      * software requirement & architecture engineering process;
      * software design and implementation engineering process;
      * software validation process;
      * software verification process;
      * software delivery and acceptance;
      * software operation process;
      * software maintenance process;
      * software management process.
2. Relationship with the system development cycle

   The SDP shall describe the phasing of the software life cycle to the system development life cycle.

   NOTE A process model representation can be used.
3. Reviews and milestones identification and associated documentation
   (a) The SDP shall describe scope and purpose of each identified review, relevant deliverables and expected outputs.
   (b) The role of involved parties or organizations at each review shall be described.

c. Software engineering standards and techniques

1. The SDP shall describe (or provide references to their description of) the applied methodologies and list the standards for each software process and relevant activity.
2. Selected methodologies and standards shall be detailed here.
3. The following items shall be covered:
   (a) software requirements analysis methods;
   (b) design methods and standards;
   (c) programming languages;
   (d) validation and verification methodology and standards;
   (e) utilization of man-machine interfaces generators;
   (f) software delivery format.

4. The requirements analysis method used shall be listed and referenced.

   NOTE Reference to applied literature or other standards can be described here.

5. Any adaptation of the requirements analysis method shall be described or referenced.
6. The selected design (architectural design and detailed design) methods shall be stated and referenced.

   NOTE Reference to applied literature or other standards can be described here.

7. Any adaptation of the design method (e.g. deviations, extensions, and avoidance of utilization of some methodology features) shall be described or referenced.
8. Any MMI standard to be applied to the software development, if code generators are utilized (e.g. constraints to be imposed to use of generators in terms of allowed specific features) shall be documented.
9. The selected software delivery format shall be described.

d. Software development and software testing environment

1. This SDP shall describe the software development environment and testing environment.
2. Hardware platforms and selected software tools to be utilized for the software development and testing shall be described and include the justification of their selection with respect to the relevant software methodology and standards.

   NOTE This covers, in particular, the tools used to configure the software for a particular mission with the parameters of the mission database.

3. The information in 2. above shall, as a minimum, include:
   (a) the compiler and cross compiler system;
   (b) the requirements analysis tool;
   (c) the tools utilized in the software image generation;
   (d) the configuration management tools;
   (e) the software design tools;
   (f) the software static analysis tools;
   (g) the software test scripts language tools;
   (h) the software testing tools (debuggers, in circuit emulator, bus analyser).

e. Software documentation plan

1. General

   The SDP shall describe or refer to all documentation relevant to the project and the documentation standards applied to the software project.

2. Software documentation identification

   The SDP, for each document to be produced (both internal documents and deliverables), shall include the documentation plan stating:
   (a) the documentation file;
   (b) the document name;

(c) the delivery requirements;

(d) the review requirements;

(e) the approval requirements.

3. Deliverable items

(a) The SDP shall list the items to be delivered.

(b) The SDF shall clearly address deliverable items internal to the software development organization (what, when and how).

4. Software documentation standards

(a) The SDP shall describe the documentation standards applicable to the project.

(b) Any tailoring to applicable documentation standards shall be detailled in this subclause.

f. ECSS standard tailoring traceability

The SDP shall include the coverage matrix of the applicable tailoring of ECSS-E-40, or provide a reference to it.

Annex K (normative)

Software product assurance plan (SPAP) DRD

K.1 DRD identification

K.1.1 Requirement identification and source document

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

NOTE Apart from potential tailoring of ECSS-Q-80 to be performed according to project specificity, ECSS-Q-80 specifies that the software product assurance plan is provided at the following milestones: SRR and PDR.

K.1.2 Purpose and objective

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

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

K.2 Expected response

K.2.1 Response identification

The requirements for project identification contained in ECSS-M-50 shall be applied to the SPAP.

K.2.2 Scope and content

The SPAP shall provide the information presented in the following sections:

<1> **Introduction**

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

<2> **Applicable and reference documents**

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

Table K-1: SPAP traceability to ECSS-E-40 Part 1 and ECSS-Q-80 subclauses

ECSS Standard	Subclauses
ECSS-Q-80B	5.2.4.1
	5.4.1.1
	5.4.1.2
	5.4.1.3
	5.4.1.4
	5.4.1.5
	5.4.1.6
	5.4.5.3
	6.1.6
	6.2.3.1
	6.2.3.3
	6.2.3.4
	6.2.5.1
	6.2.5.2
	6.2.5.3
	6.3.2.7
	6.3.2.9 expected output a
	6.3.4.1
	6.3.4.2
	7.1.1
	7.1.4
	7.1.7
	7.5.1
	7.5.2

<3> Terms, definitions and abbreviated terms

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

<4> Software product assurance programme implementation

a. Organization

1. The SPAP shall describe the organization of software product assurance activities.
2. The following topics shall be included:
   (a) organizational structure;
   (b) relationship to the other activities such as management and engineering;
   (c) relationship to the system level product assurance and safety;
   (d) independence of the software product assurance function.

b. Responsibilities

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

c. Resources

1. The SPAP shall describe the resources to be used to perform the software product assurance function.
2. The description in 1. shall include human resources and skills, hardware and software tools.

d. Reporting

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

e. Risk management

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

f. Supplier selection and control

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

g. Process assessment and improvement

1. The SPAP shall state the scope and objectives of process assessment.
2. The SPAP shall describe the methods and tools to be used for process assessment and improvement.

<5> Software process assurance

a. Software development cycle

1. The SPAP shall refer to the software development cycle description in the software development plan.
2. If not covered in the software development plan, the life cycle shall be described.

b. Projects plans

1. The SPAP shall describe all plans to be produced and used in the project.
2. The relationship between the project plans and a timely planning for their preparation and update shall be described.

c. Software dependability and safety

The SPAP shall describe the handling of critical software including the analysis to be performed and the standards applicable for critical software.

d. Software documentation and configuration management

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

e. Reuse of software

The SPAP shall describe the approach of software reuse, both covering the reuse of software developed for previous projects and the procurement and use of COTS software.

f. Product assurance planning for individual processes and activities

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

2. The following processes and activities shall be covered, taking into account the project scope and life cycle:
   (a) software requirements analysis;
   (b) software architectural design and design of software items;
   (c) coding;
   (d) testing and validation;
   (e) verification;
   (f) software delivery and acceptance.

g. Procedures et standards
   1. The SPAP shall describe or list by reference all procedures and standards applicable to the development of the software in the project.
   2. The software product assurance measures to ensure adherence to the project procedures and standards shall be described.
   3. The standards and procedures to be described or listed in accordance with 1. shall be as a minimum those covering the following aspects:
      (a) project management;
      (b) risk management;
      (c) configuration and documentation management;
      (d) verification and validation;
      (e) requirements engineering;
      (f) design;
      (g) coding;
      (h) metrication;
      (i) nonconformance control;
      (j) audits;
      (k) alerts;
      (l) procurement;
      (m) reuse of predeveloped software;
      (n) use of methods and tools;
      (o) delivery, installation and acceptance;
      (p) operations;
      (q) maintenance.

<6> Software product quality assurance

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

b. The description of the approach specified in a. shall include:
   1. the definition of quality objectives;
   2. the definition of quality model and the related quality characteristics and sub-characteristics;
   3. the specification of the product metrics, their target values and the means to collect them;
   4. the definition of a timely metrication programme;
   5. the documentation quality requirements.

<7> Compliance matrix to software product assurance requirements

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

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

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

(This page is intentionally left blank)

Bibliography

ECSS-M-10	Space project management — Project breakdown structures
ECSS-M-40B [1)]	Space project management — Configuration management

1) To be published.

(This page is intentionally left blank)

ECSS Document Improvement Proposal

1. Document I.D.	2. Document date	3. Document title
ECSS-E-40 Part 2B	31 March 2005	Software — Part 2: Document requirements definitions (DRDs)

4. Recommended improvement (identify clauses, subclauses and include modified text or graphic, attach pages as necessary)

5. Reason for recommendation

6. Originator of recommendation

Name:	Organization:	
Address:	Phone: Fax: e-mail:	**7. Date of submission:**

8. Send to ECSS Secretariat

ECSS Secretariat ESA ESTEC (TEC-QR) P.O. Box 299 2200 AG Noordwijk The Netherlands	Phone: +31-71-565-3952 Fax: +31-71-565-6839 e-mail: ECSS-Secretariat@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)