

# Software Life Cycle Process

## Working Version

Generated by: Julio Guzman

Generated with Stages

Unicon 2.11.0

Nov 6, 2024

## Table of Content

<b>1</b>	<b>Software Life Cycle Process .....</b>	<b>6</b>
1.1	Software related system requirement process (5.2) .....	6
1.1.1	Software related system requirements analysis (5.2.2) .....	7
1.1.1.1	Specification of system requirements allocated to software (5.2.2.1) .....	7
1.1.1.2	Identification of observability requirements (5.2.2.2) .....	8
1.1.1.3	Specification of HMI requirements (5.2.2.3) .....	9
1.1.2	Software related system verification (5.2.3) .....	9
1.1.2.1	Verification and validation process requirements (5.2.3.1) .....	10
1.1.2.2	System input for software validation (5.2.3.2) .....	10
1.1.2.3	System input for software installation and acceptance (5.2.3.3) .....	11
1.1.3	Software related system integration and control (5.2.4) .....	12
1.1.3.1	Identification of software versions for software integration into the system (5.2.4.1) .....	13
1.1.3.2	Supplier support to system integration (5.2.4.2) .....	13
1.1.3.3	Interface requirement specification (5.2.4.3) .....	14
1.1.3.4	System database (5.2.4.4) .....	14
1.1.3.5	Development constraints (5.2.4.5) .....	15
1.1.3.6	On board control procedures (5.2.4.6) .....	15
1.1.3.7	Development of software to be reused (5.2.4.7) .....	16
1.1.3.8	Software safety and dependability requirements (5.2.4.8) .....	16
1.1.3.9	Format and data medium (5.2.4.9) .....	17
1.1.4	System requirements review (5.2.5) .....	17
1.2	Software design & implementation engineering process (5.5) .....	18
1.2.1	Design of software items (5.5.2) .....	19
1.2.1.1	Detailed design of each software component (5.5.2.1) .....	20
1.2.1.2	Development and documentation of the software interfaces detailed design (5.5.2.2) .....	20
1.2.1.3	Production of the detailed design model (5.5.2.3) .....	21
1.2.1.4	Software detail design method (5.5.2.4) .....	21
1.2.1.5	Detailed design of real-time software (5.5.2.5) .....	22
1.2.1.6	Utilization of description techniques for the software behaviour (5.5.2.6) .....	23
1.2.1.7	Determination of design method consistency for real-time software (5.5.2.7) .....	23
1.2.1.8	Development and documentation of the software user manual (5.5.2.8) .....	23
1.2.1.9	Definition and documentation of the software unit test requirements and plan (5.5.2.9) .....	24
1.2.1.10	Conducting a detailed design review (5.5.2.10) .....	24
1.2.2	Coding and testing (5.5.3) .....	25
1.2.2.1	Development and documentation of the software units (5.5.3.1) .....	26
1.2.2.2	Software unit testing (5.5.3.2) .....	26
1.2.3	Integration (5.5.4) .....	27
1.2.3.1	Software integration test plan development (5.5.4.1) .....	27
1.2.3.2	Software units and software component integration and testing (5.5.4.2) .....	28
1.3	Software requirements & architecture engineering process (5.4) .....	29

1.3.1	Software requirements analysis (5.4.2)	30
1.3.1.1	Establishment and documentation of software requirements (5.4.2.1)	31
1.3.1.2	Definition of functional and performance requirements for in flight modification (5.4.2.2)	32
1.3.1.3	Construction of a software logical model (5.4.2.3)	33
1.3.1.4	Conducting a software requirement review (5.4.2.4)	33
1.3.2	Software architectural design (5.4.3)	35
1.3.2.1	Transformation of software requirements into a software architecture (5.4.3.1)	36
1.3.2.2	Software design method (5.4.3.2)	36
1.3.2.3	Selection of a computational model for real-time software (5.4.3.3)	37
1.3.2.4	Description of software behaviour (5.4.3.4)	37
1.3.2.5	Development and documentation of the software interfaces (5.4.3.5)	38
1.3.2.6	Definition of methods and tools for software intended for reuse (5.4.3.6)	38
1.3.2.7	Reuse of existing software (5.4.3.7)	39
1.3.2.8	Definition and documentation of the software integration requirements and plan (5.4.3.8)	39
1.3.3	Conducting a preliminary design review (5.4.4)	40
1.4	Software validation process (5.6)	41
1.4.1	Validation process implementation (5.6.2)	42
1.4.1.1	Establishment of a software validation process (5.6.2.1)	42
1.4.1.2	Selection of an ISVV organization (5.6.2.2)	43
1.4.2	Validation activities with respect to the technical specification (5.6.3)	44
1.4.2.1	Development and documentation of a software validation specification with respect to the technical specification (5.6.3.1)	44
1.4.2.2	Conducting the validation with respect to the technical specification (5.6.3.2)	45
1.4.2.3	Updating the software user manual (5.6.3.3)	45
1.4.2.4	Conducting a critical design review (5.6.3.4)	46
1.4.3	Validation activities with respect to the requirements baseline (5.6.4)	47
1.4.3.1	Development and documentation of a software validation specification with respect to the requirements baseline (5.6.4.1)	47
1.4.3.2	Conducting the validation with respect to the requirements baseline (5.6.4.2)	48
1.4.3.3	Updating the software user manual (5.6.4.3)	49
1.4.3.4	Conducting a qualification review (5.6.4.4)	49
1.5	Software delivery and acceptance process (5.7)	50
1.5.1	Software delivery and installation (5.7.2)	51
1.5.1.1	Preparation of the software product (5.7.2.1)	51
1.5.1.2	Supplier's provision of training and support (5.7.2.2)	52
1.5.1.3	Installation procedures (5.7.2.3)	52
1.5.1.4	Installation activities reporting (5.7.2.4)	53
1.5.2	Software acceptance (5.7.3)	54
1.5.2.1	Acceptance test planning (5.7.3.1)	54
1.5.2.2	Acceptance test execution (5.7.3.2)	55
1.5.2.3	Executable code generation and installation (5.7.3.3)	55
1.5.2.4	Supplier's support to customer's acceptance (5.7.3.4)	56
1.5.2.5	Evaluation of acceptance testing (5.7.3.5)	56
1.5.2.6	Conducting an acceptance review (5.7.3.6)	57
1.6	Software verification process (5.8)	57
1.6.1	Verification process implementation (5.8.2)	58

1.6.1.1	Establishment of the software verification process (5.8.2.1) .....	59
1.6.1.2	Selection of the organization responsible for conducting the verification (5.8.2.2) .....	59
1.6.2	Verification activities (5.8.3) .....	61
1.6.2.1	Verification of requirements baseline (5.8.3.1) .....	62
1.6.2.2	Verification of the technical specification (5.8.3.2) .....	63
1.6.2.3	Verification of the software architectural design (5.8.3.3) .....	63
1.6.2.4	Verification of the software detailed design (5.8.3.4) .....	64
1.6.2.5	Verification of code (5.8.3.5) .....	65
1.6.2.6	Verification of software unit testing (plan and results) (5.8.3.6) .....	66
1.6.2.7	Verification of software integration (5.8.3.7) .....	66
1.6.2.8	Verification of software validation with respect to the technical specifications and the requirements baseline (5.8.3.8) .....	67
1.6.2.9	Evaluation of validation: complementary system level validation (5.8.3.9) .....	68
1.6.2.10	Verification of software documentation (5.8.3.10) .....	68
1.6.2.11	Schedulability analysis for real-time software (5.8.3.11) .....	69
1.6.2.12	Technical budgets management (5.8.3.12) .....	69
1.6.2.13	Behaviour modelling verification (5.8.3.13) .....	70
1.7	Software management process (5.3) .....	71
1.7.1	Software life cycle management (5.3.2) .....	72
1.7.1.1	Software life cycle identification (5.3.2.1) .....	73
1.7.1.2	Identification of interfaces between development and maintenance (5.3.2.2) .....	73
1.7.1.3	Software procurement process implementation (5.3.2.3) .....	74
1.7.1.4	Automatic code generation (5.3.2.4) .....	74
1.7.1.5	Changes to baselines (5.3.2.5) .....	75
1.7.2	Joint review process (5.3.3) .....	76
1.7.2.1	Joint reviews (5.3.3.1) .....	76
1.7.2.2	Software project reviews (5.3.3.2) .....	77
1.7.2.3	Software technical reviews (5.3.3.3) .....	77
1.7.3	Software project reviews description (5.3.4) .....	79
1.7.3.1	System requirement review (5.3.4.1) .....	80
1.7.3.2	Preliminary design review (5.3.4.2) .....	81
1.7.3.3	Critical design review (5.3.4.3) .....	81
1.7.3.4	Qualification review (5.3.4.4) .....	82
1.7.3.5	Acceptance review (5.3.4.5) .....	83
1.7.4	Software technical reviews description (5.3.5) .....	83
1.7.4.1	Test readiness reviews (5.3.5.1) .....	84
1.7.4.2	Test review board (5.3.5.2) .....	84
1.7.5	Review phasing (5.3.6) .....	85
1.7.5.1	Review phasing for flight software (5.3.6.1) .....	85
1.7.5.2	Review phasing for ground software (5.3.6.2) .....	86
1.7.6	Interface management (5.3.7) .....	86
1.7.6.1	Interface management procedures (5.3.7.1) .....	87
1.7.7	Technical budget and margin management (5.3.8) .....	87
1.7.7.1	Software technical budget and margin philosophy definition (5.3.8.1) .....	88
1.7.7.2	Technical budget and margin computation (5.3.8.2) .....	88
1.7.8	Compliance to this Standard (5.3.9) .....	89

1.7.8.1	Compliance matrix (5.3.9.1) .....	89
1.7.8.2	Documentation compliance (5.3.9.2) .....	90
1.8	Software maintenance process (5.10) .....	91
1.8.1	Process implementation (5.10.2) .....	92
1.8.1.1	Establishment of the software maintenance process (5.10.2.1) .....	92
1.8.1.2	Long term maintenance for flight software (5.10.2.2) .....	93
1.8.2	Problem and modification analysis (5.10.3) .....	94
1.8.2.1	Problem analysis (5.10.3.1) .....	94
1.8.3	Modification implementation (5.10.4) .....	95
1.8.3.1	Analysis and documentation of product modification (5.10.4.1) .....	95
1.8.3.2	Documentation of software product changes (5.10.4.2) .....	96
1.8.3.3	Invoking of software engineering processes for modification implementation (5.10.4.3) .....	96
1.8.4	Conducting maintenance reviews (5.10.5) .....	97
1.8.4.1	Maintenance reviews (5.10.5.1) .....	97
1.8.4.2	Baseline for change (5.10.5.2) .....	98
1.8.5	Software migration (5.10.6) .....	99
1.8.5.1	Applicability of this Standard to software migration (5.10.6.1) .....	100
1.8.5.2	Migration planning and execution (5.10.6.2) .....	100
1.8.5.3	Contribution to the migration plan (5.10.6.3) .....	101
1.8.5.4	Preparation for migration (5.10.6.4) .....	101
1.8.5.5	Notification of transition to migrated system (5.10.6.5) .....	102
1.8.5.6	Post-operation review (5.10.6.6) .....	102
1.8.5.7	Maintenance and accessibility of data of former system (5.10.6.7) .....	103
1.8.6	Software retirement (5.10.7) .....	103
1.8.6.1	Retirement planning (5.10.7.1) .....	104
1.8.6.2	Notification of retirement plan (5.10.7.2) .....	104
1.8.6.3	Identification of requirements for software retirement (5.10.7.3) .....	105
1.8.6.4	Maintenance and accessibility to data of the retired product (5.10.7.4) .....	105
1.9	Software operation process (5.9) .....	106
1.9.1	Process implementation (5.9.2) .....	107
1.9.1.1	Operational testing definition (5.9.2.1) .....	107
1.9.1.2	Software operation support plans and procedures development (5.9.2.2) .....	108
1.9.1.3	Problem handling procedures definition (5.9.2.3) .....	108
1.9.2	Operational testing (5.9.3) .....	109
1.9.2.1	Operational testing execution (5.9.3.1) .....	109
1.9.2.2	Software operational requirements demonstration (5.9.3.2) .....	110
1.9.2.3	Software release (5.9.3.3) .....	110
1.9.3	Software operation support (5.9.4) .....	111
1.9.3.1	Software operation support performance (5.9.4.1) .....	111
1.9.3.2	Problem handling (5.9.4.2) .....	112
1.9.4	User support (5.9.5) .....	112
1.9.4.1	Assistance to the user (5.9.5.1) .....	113
1.9.4.2	Handling of user's requests (5.9.5.2) .....	113
1.9.4.3	Provisions of work-around solutions (5.9.5.3) .....	114

# 1 Software Life Cycle Process

Software engineering process model

## 1.1 Software related system requirement process (5.2)

Supplier	Inputs	Workflow	Outputs	Customer
		Software related system requirements analysis (5.2.2) +		
		Software related system verification (5.2.3) +		
		Software related system integration and control (5.2.4) +		
		System requirements review (5.2.5)		

The software related system requirement process produces the information for input to the system requirements review (SRR). This establishes the functional and the performance requirements baseline (including the interface requirement specification) (RB) of the software development. It constitutes the link between the space system processes, as defined in ECSSEST10 and ECSSEST70, and the software processes.

**Predecessor**  
None

**Successor**  

- Software verification process (5.8)

**Responsible**  

- Customer

**Inputs**  
None

**Outputs**  
None

**Guidance**  

- ECCS-E-ST-40C

**Phases**  
None

1.1.1 Software related system requirements analysis (5.2.2)

Supplier	Inputs	Workflow	Outputs	Customer
		<div>Specification of system requirements allocated to software (5.2.2.1)</div>	<div><div>Functions and performance system requirements allocated to software</div><div>Verification and validation product requirements</div><div>Software operations requirements</div><div>Software maintenance requirements</div><div>Requirements for in flight modification capabilities</div><div>Requirements for real-time</div><div>Requirements for security</div><div>Quality requirements</div></div>	
		<div>Identification of observability requirements (5.2.2.2)</div>	<div><div>System and software observability requirements</div></div>	
		<div>Specification of HMI requirements (5.2.2.3)</div>	<div><div>HMI requirements</div></div>	

<b>Predecessor</b>
None
<b>Responsible</b>
None
<b>Inputs</b>
None
<b>Guidance</b>
None

<b>Successor</b>
None
<b>Outputs</b>
None
<b>Phases</b>
None

1.1.1.1 Specification of system requirements allocated to software (5.2.2.1)

The customer shall derive system requirements allocated to software from an analysis of the specific intended use of the system, and from the results of the safety and dependability analysis

**Predecessor**

None

**Responsible**

None

**Supporting**

None

**Informed**

None

**Inputs**

None

**Successor**

None

**Accountable**

None

**Consulted**

None

**Outputs**

- Functions and performance system requirements allocated to software
- Verification and validation product requirements
- Software operations requirements
- Software maintenance requirements
- Requirements for in flight modification capabilities
- Requirements for realtime
- Requirements for security
- Quality requirements

**Guidance**

None

**Phases**

None

**1.1.1.2 Identification of observability requirements (5.2.2.2)**

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

**Predecessor**

None

**Responsible**

None

**Supporting**

None

**Informed**

None

**Inputs**

None

**Guidance**

None

**Successor**

None

**Accountable**

None

**Consulted**

None

**Outputs**

- System and software observability requirements

**Phases**

None






1.1.1.3 Specification of HMI requirements (5.2.2.3)

The customer shall specify HMI requirements, following the human factor engineering process specified in ECSSEST1011.

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"><li>HMI requirements</li></ul>
<b>Guidance</b>	<b>Phases</b>
None	None

1.1.2 Software related system verification (5.2.3)

Supplier	Inputs	Workflow	Outputs	Customer
		Verification and validation process requirements (5.2.3.1)	 Verification and validation process requirements	
		System input for software validation (5.2.3.2)	 Validation requirements and scenario	
		System input for software installation and acceptance (5.2.3.3)	 Installation and acceptance requirements at the operational and maintenance sites	

<b>Predecessor</b>
None
<b>Responsible</b>
None
<b>Inputs</b>
None
<b>Guidance</b>
None

<b>Successor</b>
None
<b>Outputs</b>
None
<b>Phases</b>
None

1.1.2.1 Verification and validation process requirements (5.2.3.1)

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

<b>Predecessor</b>
None
<b>Responsible</b>
None
<b>Supporting</b>
None
<b>Informed</b>
None
<b>Inputs</b>
None
<b>Guidance</b>
None

<b>Successor</b>
None
<b>Accountable</b>
None
<b>Consulted</b>
None
<b>Outputs</b>
<ul style="list-style-type: none"><li>• Verification and validation process requirements</li></ul>
<b>Phases</b>
None

1.1.2.2 System input for software validation (5.2.3.2)

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









<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>Validation requirements and scenario</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

### 1.1.2.3 System input for software installation and acceptance (5.2.3.3)

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>Installation and acceptance requirements at the operational and maintenance sites</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

**1.1.3 Software related system integration and control (5.2.4)**

Supplier	Inputs	Workflow	Outputs	Customer
		Identification of software versions for software integration into the system...	 Association of requirements to versions  Delivery content and media	
		Supplier support to system integration (5.2.4.2)	 System level integration support requirements	
		Interface requirement specification (5.2.4.3)	 External interface requirements specification	
		System database (5.2.4.4)	 System database content and allowed operational range	
		Development constraints (5.2.4.5)	 Design and development constraints	
		On board control procedures (5.2.4.6)	 OBCP requirements	
		Development of software to be reused (5.2.4.7)	 Requirements for 'software to be reused'	
		Software safety and dependability requirements (5.2.4.8)	 Software safety and dependability requirements	
		Format and data medium (5.2.4.9)	 Format and delivery medium of exchanged data	

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	None
<b>Guidance</b>	<b>Phases</b>
None	None

#### 1.1.3.1 Identification of software versions for software integration into the system (5.2.4.1)

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

The customer shall specify the content and media of the delivery

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>• Association of requirements to versions</li> <li>• Delivery content and media</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

#### 1.1.3.2 Supplier support to system integration (5.2.4.2)

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>System level integration support requirements</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

### 1.1.3.3 Interface requirement specification (5.2.4.3)

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>External interface requirements specification</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

### 1.1.3.4 System database (5.2.4.4)

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>System database content and allowed operational range</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

#### 1.1.3.5 Development constraints (5.2.4.5)

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>Design and development constraints</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

#### 1.1.3.6 On board control procedures (5.2.4.6)

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>OBCP requirements</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

#### 1.1.3.7 Development of software to be reused (5.2.4.7)

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>Requirements for 'software to be reused'</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

#### 1.1.3.8 Software safety and dependability requirements (5.2.4.8)

The customer shall specify the software safety and dependability requirements in accordance with ECSSQST80 clauses 5.4.4, 6.2.2 and 6.2.3, based on the results of the safety and dependability analysis performed at system level



<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>Software safety and dependability requirements</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

#### 1.1.3.9 Format and data medium (5.2.4.9)

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>Format and delivery medium of exchanged data</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

#### 1.1.4 System requirements review (5.2.5)

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

**Predecessor**

None

**Responsible**

- Customer

**Supporting**

None

**Informed**

None

**Inputs**

None

**Guidance**

None

**Successor**

None

**Accountable**

None

**Consulted**

None

**Outputs**

None

**Phases**

None

**1.2 Software design & implementation engineering process (5.5)**

Supplier	Inputs	Workflow	Outputs	Customer
		<div>Design of software items (5.5.2) +</div>		
		<div>Coding and testing (5.5.3) +</div>		
		<div>Integration (5.5.4) +</div>		

One of the outputs of this process is the detailed design of the software items identified in the software product tree (see ECSSMST10). It is provided in response to the technical specification, the ICD and the preliminary DDF. All elements of the software design are documented in the design definition file (DDF). The DDF contains all the levels of design engineering results, including software code listings.

The rationale for important design choices, and analysis and test data that show that the design meets all requirements, is added to the DJF by this process. The results of this process are the input to the critical design review (CDR).

**Predecessor**

- Software requirements & architecture engineering process (5.4)

**Successor**

- Software validation process (5.6)

**Responsible**

- Supplier

**Inputs**

None

**Outputs**

None

**Guidance**

None

**Phases**

None

**1.2.1 Design of software items (5.5.2)**

Supplier	Inputs	Workflow	Outputs	Customer
		Detailed design of each software component (5.5.2.1)	Software components design documents	
	<div> <div>Software interface control document [Preliminary]</div> <div>Preliminary internal interfaces design</div> </div>	Development and documentation of the software interfaces detailed...	<div> <div>External interfaces design (update)</div> <div>Software interface control document [Detailed]</div> <div>Internal interfaces design (update)</div> </div>	
		Production of the detailed design model (5.5.2.3)	<div> <div>Software static design model</div> <div>Software dynamic design model</div> <div>Software behavioural design model</div> </div>	
		Software detail design method (5.5.2.4)	Software design method	
		Detailed design of real-time software (5.5.2.5)	Real-time software dynamic design model	
		Utilization of description techniques for the software behaviour (5.5.2.6)	Software behavioural design model techniques	
		Determination of design method consistency for real-time software (5.5.2.7)	Compatibility of real-time design methods with the computational model	
		Development and documentation of the software user manual...	Software user manual	
		Definition and documentation of the software unit test...	Software unit test plan	
		Conducting a detailed design review (5.5.2.10)		

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	None
<b>Guidance</b>	<b>Phases</b>
None	None

#### 1.2.1.1 Detailed design of each software component (5.5.2.1)

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

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

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>Software components design documents</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

#### 1.2.1.2 Development and documentation of the software interfaces detailed design (5.5.2.2)

**Predecessor**

None

**Responsible**

None

**Supporting**

None

**Informed**

None

**Inputs**

- ICD: Software interface control document [Preliminary]
- Preliminary internal interfaces design

**Guidance**

None

**Successor**

None

**Accountable**

None

**Consulted**

None

**Outputs**

- External interfaces design (update)
- ICD: Software interface control document [Detailed]
- Internal interfaces design (update)

**Phases**

None

**1.2.1.3 Production of the detailed design model (5.5.2.3)**

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

**Predecessor**

None

**Responsible**

None

**Supporting**

None

**Informed**

None

**Inputs**

None

**Guidance**

None

**Successor**

None

**Accountable**

None

**Consulted**

None

**Outputs**

- Software static design model
- Software dynamic design model
- Software behavioural design model

**Phases**

None

**1.2.1.4 Software detail design method (5.5.2.4)**

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

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>• Software design method</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

#### 1.2.1.5 Detailed design of real-time software (5.5.2.5)

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

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

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

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

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>• Realtime software dynamic design model</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

**1.2.1.6 Utilization of description techniques for the software behaviour (5.5.2.6)**

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>Software behavioural design model techniques</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

**1.2.1.7 Determination of design method consistency for real-time software (5.5.2.7)**

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>Compatibility of realtime design methods with the computational model</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

**1.2.1.8 Development and documentation of the software user manual (5.5.2.8)**

The supplier shall develop and document the software user manual.

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>Software user manual</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

#### 1.2.1.9 Definition and documentation of the software unit test requirements and plan (5.5.2.9)

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
<ul style="list-style-type: none"> <li>Supplier</li> </ul>	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>Software unit test plan</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

#### 1.2.1.10 Conducting a detailed design review (5.5.2.10)

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



<b>Predecessor</b>
None
<b>Responsible</b>
<ul style="list-style-type: none"><li>• Supplier</li></ul>
<b>Supporting</b>
None
<b>Informed</b>
None
<b>Inputs</b>
None
<b>Guidance</b>
None

<b>Successor</b>
None
<b>Accountable</b>
None
<b>Consulted</b>
None
<b>Outputs</b>
None
<b>Phases</b>
None

1.2.2 Coding and testing (5.5.3)

Supplier	Inputs	Workflow	Outputs	Customer
		<div>Development and documentation of the software units (5.5.3.1)</div>	<div><div>Software component design document and code (update)</div><div>Source code</div><div>Software configuration file &amp; build procedures</div></div>	
		<div>Software unit testing (5.5.3.2)</div>	<div><div>Software component design document and code (update)</div><div>Source code</div><div>Software unit test plan (update)</div><div>Software unit test report</div></div>	

<b>Predecessor</b>
None
<b>Responsible</b>
None
<b>Inputs</b>
None
<b>Guidance</b>
None

<b>Successor</b>
None
<b>Outputs</b>
None
<b>Phases</b>
None

**1.2.2.1 Development and documentation of the software units (5.5.3.1)**

The supplier shall develop and document the following:

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

**Predecessor**


---

 None
**Responsible**


---

 None
**Supporting**


---

 None
**Informed**


---

 None
**Inputs**


---

 None
**Guidance**


---

 None
**Successor**


---

 None
**Accountable**


---

 None
**Consulted**


---

 None
**Outputs**

- 
- Software component design document and code (update)
  - Source code
  - Software configuration file build procedures

**Phases**


---

 None
**1.2.2.2 Software unit testing (5.5.3.2)**

The supplier shall develop and document the test procedures and data for testing each software unit.

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



The unit test shall exercise:

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

<b>Predecessor</b>
None
<b>Responsible</b>
None
<b>Supporting</b>
None
<b>Informed</b>
None
<b>Inputs</b>
None
<b>Guidance</b>
None

<b>Successor</b>
None
<b>Accountable</b>
None
<b>Consulted</b>
None
<b>Outputs</b>
<ul style="list-style-type: none"><li>• Software component design document and code (update)</li><li>• Source code</li><li>• Software unit test plan (update)</li><li>• Software unit test report</li></ul>
<b>Phases</b>
None

1.2.3 Integration (5.5.4)

Supplier	Inputs	Workflow	Outputs	Customer
		<div>Software integration test plan development (5.5.4.1)</div>	<div> Software integration test plan (update)</div>	
		<div>Software units and software component integration and testing (5.5.4.2)</div>	<div> Software integration test report</div>	

<b>Predecessor</b>
None
<b>Responsible</b>
None
<b>Inputs</b>
None
<b>Guidance</b>
None

<b>Successor</b>
None
<b>Outputs</b>
None
<b>Phases</b>
None

1.2.3.1 Software integration test plan development (5.5.4.1)

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

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

**Predecessor**

None

**Responsible**

None

**Supporting**

None

**Informed**

None

**Inputs**

None

**Guidance**

None

**Successor**

None

**Accountable**

None

**Consulted**

None

**Outputs**

- Software integration test plan (update)

**Phases**

None

**1.2.3.2 Software units and software component integration and testing (5.5.4.2)**

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

**Predecessor**

None

**Responsible**

None

**Supporting**

None

**Informed**

None

**Inputs**

None

**Guidance**

None

**Successor**

None

**Accountable**

None

**Consulted**

None

**Outputs**

- Software integration test report

**Phases**

None

### 1.3 Software requirements & architecture engineering process (5.4)

Supplier	Inputs	Workflow	Outputs	Customer
		Software requirements analysis (5.4.2) +		
		Software architectural design (5.4.3) +		
		Conducting a preliminary design review (5.4.4)		

The software requirements and architecture engineering process consists of:

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

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

During this process, the result of all significant tradeoffs, feasibility analyses, makeorbuy decisions and supporting technical assessments are documented in a design justification file (DJF).

The software requirements and architecture engineering process is completed by the preliminary design review (PDR).

#### Predecessor

None

#### Successor

- Software design & implementation engineering process (5.5)

#### Responsible

- Supplier

#### Inputs

None

#### Outputs

None

#### Guidance

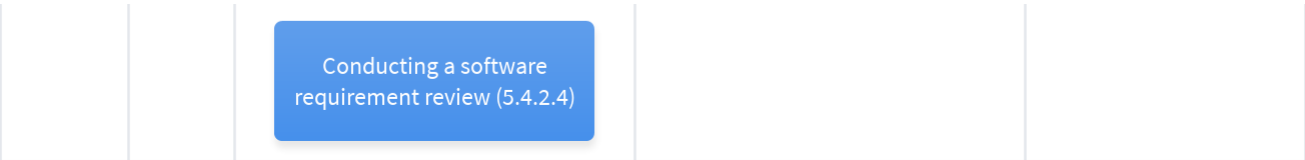
None

#### Phases

None

## 1.3.1 Software requirements analysis (5.4.2)

Supplier	Inputs	Workflow	Outputs	Customer
		<div>Establishment and documentation of software requirements (5.4.2.1)</div>	<p>Functional and performance specifications, including hardware</p> <ul style="list-style-type: none"> <li>characteristics, and environmental conditions under which the software item executes, including budgets requirements</li> <li>Operational, reliability, safety, maintainability, portability,</li> <li>configuration, delivery, adaptation and installation requirements, design constraints</li> <li>Software product quality requirements</li> <li>Security specifications, including those related to factors which can compromise sensitive information</li> <li>Human factors engineering (ergonomics including HMI usability)</li> <li>specifications, following the human factor engineering process specified in ECSS-E-ST-10-11</li> <li>Data definition and database requirements</li> <li>SRS - Validation requirements</li> <li>Reuse requirements</li> <li>ICD - Validation requirements</li> <li>Interfaces external to the software item</li> </ul>	
		<div>Definition of functional and performance requirements for in flight modification...</div>	<ul style="list-style-type: none"> <li>Specifications for in flight software modifications</li> </ul>	
		<div>Construction of a software logical model (5.4.2.3)</div>	<ul style="list-style-type: none"> <li>Software logical model</li> <li>Software logical model method</li> <li>Behavioural view in software logical model</li> </ul>	



Predecessor
None
Responsible
None
Inputs
None
Guidance
None

Successor
None
Outputs
None
Phases
None

1.3.1.1 Establishment and documentation of software requirements (5.4.2.1)

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>• Functional and performance specifications, including hardware characteristics, and environmental conditions under which the software item executes, including budgets requirements</li> <li>• Operational, reliability, safety, maintainability, portability, configuration, delivery, adaptation and installation requirements, design constraints</li> <li>• Software product quality requirements</li> <li>• Security specifications, including those related to factors which can compromise sensitive information</li> <li>• Human factors engineering (ergonomics including HMI usability) specifications, following the human factor engineering process specified in ECSSEST1011</li> <li>• Data definition and database requirements</li> <li>• SRS - Validation requirements</li> <li>• Reuse requirements</li> <li>• ICD - Validation requirements</li> <li>• Interfaces external to the software item</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

### 1.3.1.2 Definition of functional and performance requirements for in flight modification (5.4.2.2)

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



**Predecessor**

None

**Responsible**

None

**Supporting**

None

**Informed**

None

**Inputs**

None

**Guidance**

None

**Successor**

None

**Accountable**

None

**Consulted**

None

**Outputs**

- Specifications for in flight software modifications

**Phases**

None

**1.3.1.3 Construction of a software logical model (5.4.2.3)**

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

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

The logical model shall include a behavioural view.

**Predecessor**

None

**Responsible**

None

**Supporting**

None

**Informed**

None

**Inputs**

None

**Guidance**

None

**Successor**

None

**Accountable**

None

**Consulted**

None

**Outputs**

- Software logical model
- Software logical model method
- Behavioural view in software logical model

**Phases**

None

**1.3.1.4 Conducting a software requirement review (5.4.2.4)**

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

<b>Predecessor</b>
None
<b>Responsible</b>
None
<b>Supporting</b>
None
<b>Informed</b>
None
<b>Inputs</b>
None
<b>Guidance</b>
None

<b>Successor</b>
None
<b>Accountable</b>
None
<b>Consulted</b>
None
<b>Outputs</b>
None
<b>Phases</b>
None

### 1.3.2 Software architectural design (5.4.3)

Supplier	Inputs	Workflow	Outputs	Customer
		Transformation of software requirements into a software architecture (5.4.3.1)	 Software architectural design	
		Software design method (5.4.3.2)	 Software architectural design method	
		Selection of a computational model for real-time software (5.4.3.3)	 Computational model	
		Description of software behaviour (5.4.3.4)	 Software behaviour	
		Development and documentation of the software interfaces (5.4.3.5)	 Preliminary external interfaces design  Preliminary internal interfaces design	
		Definition of methods and tools for software intended for reuse (5.4.3.6)	 Software architectural design with configuration data  Software intended for reuse ☒ justification of methods and tools  Software intended for reuse ☒ evaluation of reuse potential	
		Reuse of existing software (5.4.3.7)	 Justification of reuse with respect to requirements baseline	
		Definition and documentation of the software integration...	 Software integration strategy	

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	None
<b>Guidance</b>	<b>Phases</b>
None	None

### 1.3.2.1 Transformation of software requirements into a software architecture (5.4.3.1)

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

1. describes its top-level structure;
2. identifies the software components, ensuring that all the requirements for the software item are allocated to its software components and later refined to facilitate detailed design;
3. covers as a minimum hierarchy, dependency, interfaces and operational usage for the software components;
4. documents the process, data and control aspects of the product;
5. describes the architecture static decomposition into software elements such as packages, classes or units;
6. describes the dynamic architecture, which involves the identification of active objects such as threads, tasks and processes;
7. describes the software behaviour.

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>• Software architectural design</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

### 1.3.2.2 Software design method (5.4.3.2)

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

1. software elements, their interfaces, and;
2. software elements relationships.

<b>Predecessor</b>
None
<b>Responsible</b>
None
<b>Supporting</b>
None
<b>Informed</b>
None
<b>Inputs</b>
None
<b>Guidance</b>
None

<b>Successor</b>
None
<b>Accountable</b>
None
<b>Consulted</b>
None
<b>Outputs</b>
<ul style="list-style-type: none"><li>• Software architectural design method</li></ul>
<b>Phases</b>
None

1.3.2.3 Selection of a computational model for real-time software (5.4.3.3)

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

<b>Predecessor</b>
None
<b>Responsible</b>
None
<b>Supporting</b>
None
<b>Informed</b>
None
<b>Inputs</b>
None
<b>Guidance</b>
None

<b>Successor</b>
None
<b>Accountable</b>
None
<b>Consulted</b>
None
<b>Outputs</b>
<ul style="list-style-type: none"><li>• Computational model</li></ul>
<b>Phases</b>
None

1.3.2.4 Description of software behaviour (5.4.3.4)

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>• Software behaviour</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

### 1.3.2.5 Development and documentation of the software interfaces (5.4.3.5)

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>• Preliminary external interfaces design</li> <li>• Preliminary internal interfaces design</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

### 1.3.2.6 Definition of methods and tools for software intended for reuse (5.4.3.6)

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

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

The supplier shall design the software such that mission and configuration dependant data are separated e.g. in a database.

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>• Software architectural design with configuration data</li> <li>• Software intended for reuse justification of methods and tools</li> <li>• Software intended for reuse evaluation of reuse potential</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

#### 1.3.2.7 Reuse of existing software (5.4.3.7)

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

1. identification of the reuse components and models with respect to the functional requirements baseline, and;
2. a quality evaluation of these components, applying ECSSQST80 clause 6.2.7.

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>• Justification of reuse with respect to requirements baseline</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

#### 1.3.2.8 Definition and documentation of the software integration requirements and plan (5.4.3.8)

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"><li>• Software integration strategy</li></ul>
<b>Guidance</b>	<b>Phases</b>
None	None

1.3.3 Conducting a preliminary design review (5.4.4)

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
<ul style="list-style-type: none"><li>• Supplier</li></ul>	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	None
<b>Guidance</b>	<b>Phases</b>
None	None



## 1.4 Software validation process (5.6)

Supplier	Inputs	Workflow	Outputs	Customer
		Validation process implementation (5.6.2) +		
		Validation activities with respect to the technical specification (5.6.3) +		
		Validation activities with respect to the requirements baseline (5.6.4) +		

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

The result of this process is included in the DJF.

This process is established before the PDR to basically produce a software validation plan ("process implementation").

The plan includes the validation activity with respect to the technical specification (which is held before the CDR) and the validation activity with respect to the requirements baseline (which is held before the QR and possibly repeated at AR).

This process can include a test readiness review (TRR) to verify that all test facilities and test cases and procedures are available before each significant test campaign, and under configuration control. The results relevant to the validation against the TS are checked at the CDR, and those against the RB are verified at the QR, using the DJF as input.

### Predecessor

- Software design & implementation engineering process (5.5)

### Successor

- Software delivery and acceptance process (5.7)
- Software verification process (5.8)

### Responsible

- Supplier

### Inputs

None

### Outputs

None







### Guidance

None

### Phases

None

1.4.1 Validation process implementation (5.6.2)

Supplier	Inputs	Workflow	Outputs	Customer
		<div>Establishment of a software validation process (5.6.2.1)</div>	<div><div>Software validation plan</div><div> validation process identification</div><div> Software validation plan</div><div> Software validation plan</div><div> effort and independence</div></div>	
		<div>Selection of an ISVV organization (5.6.2.2)</div>	<div><div>Independent software validation plan</div><div> organization selection</div><div>Independent software validation plan</div><div> level of independence</div></div>	

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	None
<b>Guidance</b>	<b>Phases</b>
None	None

1.4.1.1 Establishment of a software validation process (5.6.2.1)

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

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

The validation effort and the degree of organizational independence of that effort shall be determined, coherent with ECSSQST80 clause 6.3.5.19.

**Predecessor**

None

**Responsible**

None

**Supporting**

None

**Informed**

None

**Inputs**

None

**Guidance**

None

**Successor**

None

**Accountable**

None

**Consulted**

None

**Outputs**

- Software validation plan validation process identification
- Software validation plan methods and tools
- Software validation plan effort and independence

**Phases**

None

**1.4.1.2 Selection of an ISV organization (5.6.2.2)**

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

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

**Predecessor**

None

**Responsible**

None

**Supporting**

None

**Informed**

None

**Inputs**

None

**Guidance**

None

**Successor**

None

**Accountable**

None

**Consulted**

None



**Outputs**

- Independent software validation plan organization selection
- Independent software validation plan level of independence

**Phases**

None

### 1.4.2 Validation activities with respect to the technical specification (5.6.3)

Supplier	Inputs	Workflow	Outputs	Customer
		Development and documentation of a software validation specification with...	 Software validation specification with respect to the technical specification	
		Conducting the validation with respect to the technical specification (5.6.3.2)	 Software validation specification with respect to TS	
		Updating the software user manual (5.6.3.3)	 Software user manual (update)	
		Conducting a critical design review (5.6.3.4)		

#### Predecessor

None

#### Successor

None

#### Responsible

None

#### Inputs

None

#### Outputs

None

#### Guidance

None

#### Phases

None

#### 1.4.2.1 Development and documentation of a software validation specification with respect to the technical specification (5.6.3.1)

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

1. testing with stress, boundary, and singular inputs;
2. testing the software product for its ability to isolate and reduce the effect of errors;
3. testing that the software product can perform successfully in a representative operational environment;
4. external interface testing including boundaries, protocols and timing test;
5. testing HMI applications as per ECSSEST1011.

Validation shall be performed by test.

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"><li>Software validation specification with respect to the technical specification</li></ul>
<b>Guidance</b>	<b>Phases</b>
None	None

1.4.2.2 Conducting the validation with respect to the technical specification (5.6.3.2)

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"><li>SVS: Software validation specification with respect to TS</li></ul>
<b>Guidance</b>	<b>Phases</b>
None	None

1.4.2.3 Updating the software user manual (5.6.3.3)

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

<b>Predecessor</b>
None
<b>Responsible</b>
None
<b>Supporting</b>
None
<b>Informed</b>
None
<b>Inputs</b>
None
<b>Guidance</b>
None

<b>Successor</b>
None
<b>Accountable</b>
None
<b>Consulted</b>
None
<b>Outputs</b>
<ul style="list-style-type: none"><li>• Software user manual (update)</li></ul>
<b>Phases</b>
None




1.4.2.4 Conducting a critical design review (5.6.3.4)

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

<b>Predecessor</b>
None
<b>Responsible</b>
<ul style="list-style-type: none"><li>• Supplier</li></ul>
<b>Supporting</b>
None
<b>Informed</b>
None
<b>Inputs</b>
None
<b>Guidance</b>
None

<b>Successor</b>
None
<b>Accountable</b>
None
<b>Consulted</b>
None
<b>Outputs</b>
None
<b>Phases</b>
None

### 1.4.3 Validation activities with respect to the requirements baseline (5.6.4)

Supplier	Inputs	Workflow	Outputs	Customer
		Development and documentation of a software validation specification with...	 Software validation specification with respect to RB	
		Conducting the validation with respect to the requirements baseline...	 Software validation report with respect to RB	
		Updating the software user manual (5.6.4.3)	 Software user manual (update)	
		Conducting a qualification review (5.6.4.4)		

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

#### Predecessor

None

#### Successor

None

#### Responsible

None

#### Inputs

None

#### Outputs

None

#### Guidance

None

#### Phases

None

#### 1.4.3.1 Development and documentation of a software validation specification with respect to the requirements baseline (5.6.4.1)

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

1. testing against the mission data and scenario specified by the customer in 5.2.3.1
2. testing with stress, boundary, and singular inputs;
3. testing the software product for its ability to isolate and reduce the effect of errors;
4. testing that the software product can perform successfully in a representative operational and nonintrusive environment.
5. external interface testing including boundaries, protocols and timing test;
6. testing HMI applications as per ECSSEST1011.

Validation shall be performed by test.

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"><li>SVS: Software validation specification with respect to RB</li></ul>
<b>Guidance</b>	<b>Phases</b>
None	None

1.4.3.2 Conducting the validation with respect to the requirements baseline (5.6.4.2)

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

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"><li>Software validation report with respect to RB</li></ul>
<b>Guidance</b>	<b>Phases</b>
None	None



1.4.3.3 Updating the software user manual (5.6.4.3)

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"><li>Software user manual (update)</li></ul>
<b>Guidance</b>	<b>Phases</b>
None	None

1.4.3.4 Conducting a qualification review (5.6.4.4)

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	None
<b>Guidance</b>	<b>Phases</b>
None	None

1.5 Software delivery and acceptance process (5.7)

Supplier	Inputs	Workflow	Outputs	Customer
		<div>Software delivery and installation (5.7.2)<div></div></div>		
		<div>Software acceptance (5.7.3)<div></div></div>		

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

**Predecessor**

- Software validation process (5.6)

**Successor**

- Software verification process (5.8)

**Responsible**

- Customer
- Supplier

**Inputs**

None

**Outputs**

None

**Guidance**

None

**Phases**

None

1.5.1 Software delivery and installation (5.7.2)

Supplier	Inputs	Workflow	Outputs	Customer
		Preparation of the software product (5.7.2.1)	<div>Software release document</div> <div>Software product and media labels</div>	
		Supplier's provision of training and support (5.7.2.2)	<div>Training material</div>	
		Installation procedures (5.7.2.3)	<div>Installation procedures</div>	
		Installation activities reporting (5.7.2.4)	<div>Installation report</div>	

Predecessor
None
Responsible
None
Inputs
None
Guidance
None

Successor
None
Outputs
None
Phases
None

1.5.1.1 Preparation of the software product (5.7.2.1)

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

Predecessor
None
Responsible
None
Supporting
None
Informed
None

Successor
None
Accountable
None
Consulted
None

**Inputs**

None

**Outputs**

- Software release document
- Software product and media labels

**Guidance**

None

**Phases**

None

**1.5.1.2 Supplier's provision of training and support (5.7.2.2)**

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

**Predecessor**

None

**Successor**

None

**Responsible**

None

**Accountable**

None

**Supporting**

None

**Consulted**

None

**Informed**

None

**Inputs**

None

**Outputs**

- Training material

**Guidance**

None

**Phases**

None

**1.5.1.3 Installation procedures (5.7.2.3)**

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

**Predecessor**

None

**Successor**

None

**Responsible**

None

**Accountable**

None

**Supporting**

None

**Consulted**

None

**Informed**

None

**Inputs**

None

**Outputs**

- Installation procedures

**Guidance****Phases**

None

None

1.5.1.4 Installation activities reporting (5.7.2.4)

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

The supplier shall assist the customer with the set-up activities.

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

The installation events and results shall be documented.

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"><li>Installation report</li></ul>
<b>Guidance</b>	<b>Phases</b>
None	None

1.5.2 Software acceptance (5.7.3)

Supplier	Inputs	Workflow	Outputs	Customer
		Acceptance test planning (5.7.3.1)	 Acceptance test plan	
		Acceptance test execution (5.7.3.2)	 Acceptance test report	
		Executable code generation and installation (5.7.3.3)	 Software product and media labels	
		Supplier's support to customer's acceptance (5.7.3.4)	 DJF - Joint review reports	
		Evaluation of acceptance testing (5.7.3.5)	 Traceability of acceptance tests to the requirements baseline	
		Conducting an acceptance review (5.7.3.6)		

<b>Predecessor</b>
None
<b>Responsible</b>
None
<b>Inputs</b>
None
<b>Guidance</b>
None

<b>Successor</b>
None
<b>Outputs</b>
None
<b>Phases</b>
None

1.5.2.1 Acceptance test planning (5.7.3.1)

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

<b>Predecessor</b>	<b>Successor</b>
--------------------	------------------

None
<b>Responsible</b>
None
<b>Supporting</b>
None
<b>Informed</b>
None
<b>Inputs</b>
None
<b>Guidance</b>
None

None
<b>Accountable</b>
None
<b>Consulted</b>
None
<b>Outputs</b>
<ul style="list-style-type: none"><li>Acceptance test plan</li></ul>
<b>Phases</b>
None

1.5.2.2 Acceptance test execution (5.7.3.2)

The customer shall perform the acceptance testing

<b>Predecessor</b>
None
<b>Responsible</b>
None
<b>Supporting</b>
None
<b>Informed</b>
None
<b>Inputs</b>
None
<b>Guidance</b>
None

<b>Successor</b>
None
<b>Accountable</b>
None
<b>Consulted</b>
None
<b>Outputs</b>
<ul style="list-style-type: none"><li>Acceptance test report</li></ul>
<b>Phases</b>
None

1.5.2.3 Executable code generation and installation (5.7.3.3)

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

<b>Predecessor</b>
None
<b>Responsible</b>
None
<b>Supporting</b>
None

<b>Successor</b>
None
<b>Accountable</b>
None
<b>Consulted</b>
None

**Informed**

None

**Inputs**

None

**Guidance**

None

**Outputs**

- Software product and media labels

**Phases**

None

**1.5.2.4 Supplier's support to customer's acceptance (5.7.3.4)**

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

**Predecessor**

None

**Responsible**

None

**Supporting**

None

**Informed**

None

**Inputs**

None

**Guidance**

None

**Successor**

None

**Accountable**

None

**Consulted**

None

**Outputs**

- DJF - Joint review reports

**Phases**

None

**1.5.2.5 Evaluation of acceptance testing (5.7.3.5)**

The acceptance tests shall be traced to the requirements baseline

**Predecessor**

None

**Responsible**

None

**Supporting**

None

**Informed**

None

**Inputs**

None

**Successor**

None

**Accountable**

None

**Consulted**

None

**Outputs**

- Traceability of acceptance tests to the requirements baseline



Guidance

None

Phases

None

1.5.2.6 Conducting an acceptance review (5.7.3.6)

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

Predecessor

None

Successor

None

Responsible

None

Accountable

None

Supporting

None

Consulted

None

Informed

None

Inputs

None

Outputs

None

Guidance

None

Phases

None

1.6 Software verification process (5.8)

Supplier	Inputs	Workflow	Outputs	Customer
		<div>Verification process implementation (5.8.2) +</div>		
		<div>Verification activities (5.8.3) +</div>		

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

This process is concurrent with all the previous processes.

The customer verifies the requirement baseline for the SRR.

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

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

<b>Predecessor</b>
<ul style="list-style-type: none"><li>• Software related system requirement process (5.2)</li><li>• Software validation process (5.6)</li><li>• Software delivery and acceptance process (5.7)</li></ul>
<b>Responsible</b>
<ul style="list-style-type: none"><li>• Supplier</li></ul>
<b>Inputs</b>
None
<b>Guidance</b>
None

<b>Successor</b>
<ul style="list-style-type: none"><li>• Software management process (5.3)</li></ul>
<b>Outputs</b>
None
<b>Phases</b>
None

1.6.1 Verification process implementation (5.8.2)

Supplier	Inputs	Workflow	Outputs	Customer
		<div>Establishment of the software verification process (5.8.2.1)</div>	<div>Software verification plan ☒ verification process identification</div> <div>Software verification plan ☒ software products identification</div> <div>Software verification plan ☒ activities, methods and tools</div> <div>Software verification plan ☒ organizational independence, risk and effort identification</div>	
		<div>Selection of the organization responsible for conducting the verification (5.8.2.2)</div>	<div>Independent software validation plan ☒ organization selection</div> <div>Independent software validation plan ☒ level of independence</div>	

<b>Predecessor</b>
None
<b>Responsible</b>
None
<b>Inputs</b>
None
<b>Guidance</b>
None

<b>Successor</b>
None
<b>Outputs</b>
None
<b>Phases</b>
None

**1.6.1.1 Establishment of the software verification process (5.8.2.1)**

1. The verification process shall be established by the supplier to verify the software products
2. Life cycle activities and software products needing verification shall be determined based upon the scope, magnitude, complexity, and criticality analysis.
3. Verification activities and tasks defined in clause 5.8.3, including associated methods, techniques, and tools for performing the tasks, shall be selected for the life cycle activities and software products.
4. A determination shall be made concerning the verification effort, the identification of risks and the degree of organizational independence.

**Predecessor**

None

**Responsible**

None

**Supporting**

None

**Informed**

None

**Inputs**

None

**Successor**

None

**Accountable**

None

**Consulted**

None

**Outputs**

- Software verification plan verification process identification
- Software verification plan software products identification
- Software verification plan activities, methods and tools
- Software verification plan organizational independence, risk and effort identification

**Guidance**

None

**Phases**

None

**1.6.1.2 Selection of the organization responsible for conducting the verification (5.8.2.2)**

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

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

**Predecessor**

None

**Responsible**

None

**Supporting**

None

**Successor**

None

**Accountable**

None

**Consulted**











None

<b>Informed</b>
None
<b>Inputs</b>
None
<b>Guidance</b>
None

<b>Outputs</b>
<ul style="list-style-type: none"><li>• Independent software validation plan organization selection</li><li>• Independent software validation plan level of independence</li></ul>
<b>Phases</b>
None

## 1.6.2 Verification activities (5.8.3)

Supplier	Inputs	Workflow	Outputs	Customer
		Verification of requirements baseline (5.8.3.1)	 Requirements baseline verification report	
		Verification of the technical specification (5.8.3.2)		
		Verification of the software architectural design (5.8.3.3)	 Software architectural design and interface verification report	
		Verification of the software detailed design (5.8.3.4)	 Detailed design verification report	
		Verification of code (5.8.3.5)	 Software code traceability matrices  Software code verification report  Code coverage verification report  Robustness verification report	
		Verification of software unit testing (plan and results) (5.8.3.6)	 Software unit tests traceability matrices  Software unit testing verification report	
		Verification of software integration (5.8.3.7)	 Software integration verification report	
		Verification of software validation with respect to the technical specifications and...	 Validation report evaluation with respect to the technical specification  Validation report evaluation with respect to the requirements baseline	
		complementary system level validation (5.8.3.9)	 Complement of validation at system level	

		Verification of software documentation (5.8.3.10)	 Software documentation verification report	
		Schedulability analysis for real-time software (5.8.3.11)	 Schedulability analysis  Schedulability analysis (update)	
		Technical budgets management (5.8.3.12)	 Technical budgets  memory and CPU estimation  Technical budgets (update)  memory and CPU estimation  Technical budgets (update)  memory and CPU calculation	
		Behaviour modelling verification (5.8.3.13 )	 Software behaviour verification	

Predecessor
None
Responsible
None
Inputs
None
Guidance
None

Successor
None
Outputs
None
Phases
None

1.6.2.1 Verification of requirements baseline (5.8.3.1)

Predecessor
None
Responsible
None
Supporting
None
Informed
None
Inputs
None

Successor
None
Accountable
None
Consulted
None
Outputs
<ul style="list-style-type: none"><li>Requirements baseline verification report</li></ul>

**Guidance**

None

**Phases**

None

**1.6.2.2 Verification of the technical specification (5.8.3.2)**

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

1. software requirements and interface are externally and internally consistent (not implying formal proof consistency);
2. the traceability between system requirements and software requirements is complete;
3. the software requirements that are not traced to the system requirements allocated to software are justified;
4. software requirements are verifiable;
5. software design is feasible;
6. operations and maintenance are feasible;
7. the software requirements related to safety, security, and criticality are correct;
8. the hardware environment constraints are identified;
9. the implementation constraints are identified;
10. the requirement verification method as specified in ECSS QST80 clause 7.2.1.3 is feasible.
11. the logical model has been checked;

**Predecessor**

None

**Successor**

None

**Responsible**

None

**Accountable**

None

**Supporting**

None

**Consulted**

None

**Informed**

None

**Inputs**

None

**Outputs**

None

**Guidance**

None

**Phases**

None

**1.6.2.3 Verification of the software architectural design (5.8.3.3)**

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

1. architecture and interface are externally consistent with the requirements of the software item;
2. there is internal consistency between the software components;
3. the traceability between the requirements and the software components is complete;
4. the software components that are not traced to the software requirements are justified;
5. producing a detailed design is feasible;
6. operations and maintenance are feasible;
7. the design is correct with respect to the requirements and the interfaces, including safety, security and other critical requirements;
8. the design implements proper sequence of events, inputs, outputs, interfaces, logic flow, allocation of timing and sizing budgets, and error handling;

9. the hierarchical breakdown from high level components to terminal ones is provided;
10. the dynamic features (tasks definition and priorities, synchronization mechanisms, shared resources management) are provided and the realtime choices are justified;
11. the synchronisation between external interface and internal timing is achieved.

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>Software architectural design and interface verification report</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

#### 1.6.2.4 Verification of the software detailed design (5.8.3.4)

The supplier shall verify the software detailed design ensuring that:

1. detailed design is externally consistent with the architecture;
2. there is internal consistency between software components and software units;
3. the traceability between the architecture and the detailed design is complete;
4. the software units that are not traced to the components are justified;
5. testing is feasible, by assessing that:
  - (a) commandability and observability features are identified and included in the detailed design in order to prepare the effective testing of the performance requirements;
  - (b) computational invariant properties and temporal properties are added within the design;
  - (c) fault injection is possible.
6. operation and maintenance are feasible;
7. the design is correct with respect to requirements and interfaces, including safety, security, and other critical requirements;
8. the design implements proper sequence of events, inputs, outputs, interfaces, logic flow, allocation of timing and sizing budgets, and error handling;
9. the design model has been checked;
10. the hierarchical breakdown from high level components to terminal ones is provided;
11. the dynamic features (tasks definition and priorities, synchronization mechanisms, shared resources management) are provided and the realtime choices are justified;
12. the synchronisation between external interface and internal timing is achieved;

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>



None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>Detailed design verification report</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

#### 1.6.2.5 Verification of code (5.8.3.5)

The supplier shall verify the software code ensuring that:

1. the code is externally consistent with the requirements and design of the software item;
2. there is internal consistency between software units;
3. the code is traceable to design and requirements, testable, correct, and in conformity to software requirements and coding standards;
4. the code that is not traced to the units is justified;
5. the code implements proper events sequences, consistent interfaces, correct data and control flow, completeness, appropriate allocation of timing and sizing budgets, and error handling;
6. the code implements safety, security, and other critical requirements correctly as shown by appropriate methods;
7. the effects of run-time errors are controlled;
8. there are no memory leaks;
9. numerical protection mechanisms are implemented.

Code coverage shall be measured by analysis of the results of the execution of tests

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

In case the traceability between source code and object code cannot be verified (e.g. use of compiler optimization), the supplier shall perform additional code coverage analysis on object code level

The supplier shall verify source code robustness (e.g. resource sharing, division by zero, pointers, runtime errors).

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>Software code traceability matrices</li> </ul>

- Software code verification report
- Code coverage verification report
- Robustness verification report

**Guidance**

None

**Phases**

None

**1.6.2.6 Verification of software unit testing (plan and results) (5.8.3.6)**

The supplier shall verify the unit tests results ensuring that:

1. the unit tests are consistent with detailed design and requirements;
2. the unit tests are traceable to software requirements, design and code;

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

3. software integration and testing are feasible;
4. operation and maintenance are feasible;
5. all activities defined in clause 5.5.3 are performed;
6. test results conform to expected results;
7. test results, test logs, test data, test cases and procedures, and test documentation are maintained under configuration management;
8. normal termination (i.e. the test end criteria defined in the unit test plan) is achieved;
9. abnormal termination of testing process (e.g. incorrect major fault, out of time) is reported;
10. abnormal termination condition is documented in summary section of the unit test report, together with the unfinished testing and any uncorrected faults.

**Predecessor**

None

**Successor**

None

**Responsible**

None

**Accountable**

None

**Supporting**

None

**Consulted**

None

**Informed**

None

**Inputs**

None

**Outputs**

- Software unit tests traceability matrices
- Software unit testing verification report

**Guidance**

None

**Phases**

None

**1.6.2.7 Verification of software integration (5.8.3.7)**

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

1. traceability to software architectural design;
2. internal consistency;
3. interface testing goals;
4. conformance to expected results.

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>Software integration verification report</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

#### 1.6.2.8 Verification of software validation with respect to the technical specifications and the requirements baseline (5.8.3.8)

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

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>Validation report evaluation with respect to the technical specification</li> <li>Validation report evaluation with respect to the requirements baseline</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

**1.6.2.9 Evaluation of validation: complementary system level validation (5.8.3.9)**

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

**Predecessor**

None

**Responsible**

None

**Supporting**

None

**Informed**

None

**Inputs**

None

**Guidance**

None

**Successor**

None

**Accountable**

None

**Consulted**

None

**Outputs**

- Complement of validation at system level

**Phases**

None

**1.6.2.10 Verification of software documentation (5.8.3.10)**

The supplier shall verify the software documentation ensuring that:

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

**Predecessor**

None

**Responsible**

None

**Supporting**

None

**Informed**

None

**Inputs**

None

**Guidance**

None

**Successor**

None

**Accountable**

None

**Consulted**

None

**Outputs**

- Software documentation verification report

**Phases**

None

### 1.6.2.11 Schedulability analysis for real-time software (5.8.3.11)

As part of the verification of the software requirements and architectural design , the supplier shall use an analytical model (or use modelling and simulation if it can be demonstrated that no analytical model exists) to perform a schedulability analysis and prove that the design is feasible.

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

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>Schedulability analysis</li> <li>Schedulability analysis (update)</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

### 1.6.2.12 Technical budgets management (5.8.3.12)

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

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

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	

**Inputs**

None

**Outputs**

- Technical budgets memory and CPU estimation
- Technical budgets (update) memory and CPU estimation
- Technical budgets (update) memory and CPU calculation

**Guidance**

None

**Phases**

None

**1.6.2.13 Behaviour modelling verification (5.8.3.13 )**

As support to the verification of the software requirements, the supplier shall verify the software behaviour using the behavioural view of the logical model produced in 5.4.2.3c.

As support to the verification of the software architectural design, the supplier shall verify the software behaviour using the behavioural view of the architecture produced in clause 5.4.3.4

c. As support to the verification of the software detailed design, the supplier shall verify the software behaviour using the software behavioural design model produced in 5.5.2.3a. eo c., by means of the techniques defined in 5.5.2.6.

**Predecessor**

None

**Successor**

None

**Responsible**

None

**Accountable**

None

**Supporting**

None

**Consulted**

None

**Informed**

None

**Inputs**

None

**Outputs**

- Software behaviour verification

**Guidance**

None

**Phases**

None

## 1.7 Software management process (5.3)

Supplier	Inputs	Workflow	Outputs	Customer
		Software life cycle management (5.3.2) +		
		Joint review process (5.3.3) +		
		Software project reviews description (5.3.4) +		
		Software technical reviews description (5.3.5) +		
		Review phasing (5.3.6) +		
		Interface management (5.3.7) +		
		Technical budget and margin management (5.3.8) +		
		Compliance to this Standard (5.3.9) +		

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

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

The reviews are defined and positioned in the life cycle in clause 5.3. They are called in their relevant process in clause 5.2, 5.4, 5.5, 5.6, and 5.7.

### Predecessor

- Software verification process (5.8)

### Successor

- Software maintenance process (5.10)

**Responsible**

- Customer
- Supplier

**Inputs**

None

**Guidance**

- AR - Acceptance Review

**Outputs**

None

**Phases**

None

**1.7.1 Software life cycle management (5.3.2)**

Supplier	Inputs	Workflow	Outputs	Customer
		Software life cycle identification (5.3.2.1)	<ul style="list-style-type: none"> <li>Software life cycle definition</li> <li>Development strategy, standards, techniques, development and testing environment</li> </ul>	
		Identification of interfaces between development and maintenance (5.3.2.2)	<ul style="list-style-type: none"> <li>Identification of interface between development and maintenance</li> </ul>	
		Software procurement process implementation (5.3.2.3)	<ul style="list-style-type: none"> <li>Software procurement process documentation and implementation</li> </ul>	
		Automatic code generation (5.3.2.4)	<ul style="list-style-type: none"> <li>Autocode input model review</li> <li>Autocode interface definition and resource allocation</li> <li>Automatic code generation development process and tools</li> <li>Automatic code generation verification and validation strategy</li> <li>Automatic code generation configuration management</li> </ul>	
		Changes to baselines (5.3.2.5)	<ul style="list-style-type: none"> <li>Changes to baselines procedures</li> </ul>	

The supplier shall document and implement the software procurement process as specified in ECSSQST80 clause 5.5.



**Predecessor**

None

**Responsible**

None

**Inputs**

None

**Guidance**

None

**Successor**

None

**Outputs**

None

**Phases**

None

**1.7.1.1 Software life cycle identification (5.3.2.1)**

The software supplier shall define and follow a software life cycle including phases, their inputs and outputs, and joint reviews, in accordance with the overall project constraints and with ECSSMST10.

The life cycle shall be chosen, assessing the specifics of the project technical approaches and the relevant project risks. The software supplier shall define the development strategy, the software engineering standards and techniques, the software development and the software testing environment.

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

**Predecessor**

None

**Responsible**

None

**Supporting**

None

**Informed**

None

**Inputs**

None

**Guidance**

None

**Successor**

None

**Accountable**

None

**Consulted**

None

**Outputs**

- Software life cycle definition
- Development strategy, standards, techniques, development and testing environment

**Phases**

None

**1.7.1.2 Identification of interfaces between development and maintenance (5.3.2.2)**

The interfaces between development and maintenance (e.g. documents to be handed over, tools to be kept for maintenance) shall be identified in the software life cycle

**Predecessor**

None

**Successor**

None

**Responsible**

None

**Supporting**

None

**Informed**

None

**Inputs**

None

**Guidance**

None

**Accountable**

None

**Consulted**

None

**Outputs**

- Identification of interface between development and maintenance

**Phases**

None

**1.7.1.3 Software procurement process implementation (5.3.2.3)****Predecessor**

None

**Responsible**

None

**Supporting**

None

**Informed**

None

**Inputs**

None

**Guidance**

None

**Successor**

None

**Accountable**

None

**Consulted**

None

**Outputs**

- Software procurement process documentation and implementation

**Phases**

None

**1.7.1.4 Automatic code generation (5.3.2.4)**

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

In the case of coexisting autcoded and manually written parts, the software development plan shall include the definition of a clear interface definition and resource allocation (memory, CPU) at PDR.

The input model management, the code generation process and supporting tools shall be documented in the SDP.

The supplier shall define in the SDP the verification and validation strategy for automatic code generation as a result of the trade off between the qualification of the code generation toolchain and the end to end validation strategy of the software item, or any combination thereof, in relation with ECSSQST80 clause 6.2.8.

The configuration management of the automatic code generation related elements shall be defined in the SCMP.

**Predecessor**

None

**Responsible**

None

**Supporting**

None

**Informed**

None

**Inputs**

None

**Successor**

None

**Accountable**

None

**Consulted**

None

**Outputs**

- Autocode input model review
- Autocode interface definition and resource allocation
- Automatic code generation development process and tools
- Automatic code generation verification and validation strategy
- Automatic code generation configuration management

**Guidance**

None

**Phases**

None

**1.7.1.5 Changes to baselines (5.3.2.5)**

Changes to baselines shall be handled by the configuration management process described in clause 6.2.4 of ECSSQST80.

**Predecessor**

None

**Responsible**

None

**Supporting**

None

**Informed**

None

**Inputs**

None

**Guidance**

None

**Successor**

None

**Accountable**

None

**Consulted**

None







**Outputs**

- Changes to baselines procedures

**Phases**

None

### 1.7.2 Joint review process (5.3.3)

Supplier	Inputs	Workflow	Outputs	Customer
		Joint reviews (5.3.3.1)	 DJF - Joint review reports	
		Software project reviews (5.3.3.2)	 Software project reviews included in the software life cycle definition  Review Plan	
		Software technical reviews (5.3.3.3)	 Software technical reviews included in the software life cycle definition  Technical reviews process  Software technical reviews included in the software life cycle definition	

#### Predecessor

None

#### Successor

None

#### Responsible

None

#### Inputs

None

#### Outputs

None

#### Guidance

None

#### Phases

None

#### 1.7.2.1 Joint reviews (5.3.3.1)

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

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>DJF - Joint review reports</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

#### 1.7.2.2 Software project reviews (5.3.3.2)

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

The review process specified in ECSSMST1001 shall apply to all software project reviews, including the agreement on a review plan before the review process is started.

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>Software project reviews included in the software life cycle definition</li> <li>Review Plan</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

#### 1.7.2.3 Software technical reviews (5.3.3.3)

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

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

The supplier shall use the software technical reviews to implement intermediate reviews, in particular for incremental life cycle.

**Predecessor**  
None

**Responsible**  
None

**Supporting**  
None

**Informed**  
None

**Inputs**  
None

**Guidance**  
None

**Successor**  
None

**Accountable**  
None

**Consulted**  
None

**Outputs**

- Software technical reviews included in the software life cycle definition
- Technical reviews process
- Software technical reviews included in the software life cycle definition

**Phases**  
None

## 1.7.3 Software project reviews description (5.3.4)

Supplier	Inputs	Workflow	Outputs	Customer
		System requirement review (5.3.4.1)	 Approved requirements baseline	
		Preliminary design review (5.3.4.2)	 TS - Approved technical specification and interface, architecture and plans  DDF - Approved technical specification and interface, architecture and plans  DJF - Approved technical specification and interface, architecture and plans  MGT - Approved technical specification and interface, architecture and plans	
		Critical design review (5.3.4.3)	 DDF - Approved design definition file and design justification file  DDF - Approved detailed design, interface design and budget  DJF - Approved design definition file and design justification file  DJF - Approved detailed design, interface design and budget	
		Qualification review (5.3.4.4)	 RB - Qualified software product  TS - Qualified software product  DDF - Qualified software product  DJF - Qualified software product  MGT - Qualified software product  MF - Qualified software product	
			 RB - Accepted software product  TS - Accepted software	

		<div>Acceptance review (5.3.4.5)</div>	<div><div>product</div><div>DDF - Accepted software product</div><div>DJF - Accepted software product</div><div>MGT - Accepted software product</div><div>MF - Accepted software product</div></div>	
--	--	--	--	--

<b>Predecessor</b>
None
<b>Responsible</b>
None
<b>Inputs</b>
None
<b>Guidance</b>
None

<b>Successor</b>
None
<b>Outputs</b>
None
<b>Phases</b>
None

1.7.3.1 System requirement review (5.3.4.1)

After completion of the software requirements baseline specification, a system requirements review (SRR) shall take place.  
AIM: Reach the approval of the software requirements baseline by all stakeholders.

<b>Predecessor</b>
None
<b>Responsible</b>
None
<b>Supporting</b>
None
<b>Informed</b>
None
<b>Inputs</b>
None
<b>Guidance</b>
None

<b>Successor</b>
None
<b>Accountable</b>
None
<b>Consulted</b>
None
<b>Outputs</b>
<div>• Approved requirements baseline</div>
<b>Phases</b>
None



### 1.7.3.2 Preliminary design review (5.3.4.2)

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

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

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

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

#### Predecessor

None

#### Responsible

None

#### Supporting

None

#### Informed

None

#### Inputs

None

#### Successor

None

#### Accountable

None

#### Consulted

None

#### Outputs

- TS -Approved technical specification and interface, architecture and plans
- DDF - Approved technical specification and interface, architecture and plans
- DJF - Approved technical specification and interface, architecture and plans
- MGT - Approved technical specification and interface, architecture and plans

#### Guidance

None

#### Phases

None

### 1.7.3.3 Critical design review (5.3.4.3)

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

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

#### Predecessor

None

#### Responsible

#### Successor

None

#### Accountable

None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>• DDF - Approved design definition file and design justification file</li> <li>• DDF - Approved detailed design, interface design and budget</li> <li>• DJF - Approved design definition file and design justification file</li> <li>• DJF - Approved detailed design, interface design and budget</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

#### 1.7.3.4 Qualification review (5.3.4.4)

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>• RB - Qualified software product</li> <li>• TS - Qualified software product</li> <li>• DDF - Qualified software product</li> <li>• DJF - Qualified software product</li> <li>• MGT - Qualified software product</li> <li>• MF - Qualified software product</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None



1.7.3.5 Acceptance review (5.3.4.5)

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

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"><li>• RB - Accepted software product</li><li>• TS - Accepted software product</li><li>• DDF - Accepted software product</li><li>• DJF - Accepted software product</li><li>• MGT - Accepted software product</li><li>• MF - Accepted software product</li></ul>
<b>Guidance</b>	<b>Phases</b>
None	None

1.7.4 Software technical reviews description (5.3.5)

Supplier	Inputs	Workflow	Outputs	Customer
		<div>Test readiness reviews (5.3.5.1)</div>	<div> Confirmation of readiness of test activities</div>	
		<div>Test review board (5.3.5.2)</div>	<div> Approved test results</div>	

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	
None	
<b>Inputs</b>	<b>Outputs</b>

None
<b>Guidance</b>
None

None
<b>Phases</b>
None

1.7.4.1 Test readiness reviews (5.3.5.1)

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

<b>Predecessor</b>
None
<b>Responsible</b>
None
<b>Supporting</b>
None
<b>Informed</b>
None
<b>Inputs</b>
None
<b>Guidance</b>
None

<b>Successor</b>
None
<b>Accountable</b>
None
<b>Consulted</b>
None
<b>Outputs</b>
<ul style="list-style-type: none"><li>Confirmation of readiness of test activities</li></ul>
<b>Phases</b>
None



1.7.4.2 Test review board (5.3.5.2)

The test review board (TRB) shall approve test results at the end of test activities, as defined in the software development plan.

<b>Predecessor</b>
None
<b>Responsible</b>
None
<b>Supporting</b>
None
<b>Informed</b>
None
<b>Inputs</b>
None
<b>Guidance</b>
None

<b>Successor</b>
None
<b>Accountable</b>
None
<b>Consulted</b>
None
<b>Outputs</b>
<ul style="list-style-type: none"><li>Approved test results</li></ul>
<b>Phases</b>
None

### 1.7.5 Review phasing (5.3.6)

Supplier	Inputs	Workflow	Outputs	Customer
		Review phasing for flight software (5.3.6.1)	 Flight software review phasing	
		Review phasing for ground software (5.3.6.2)	 Ground software review phasing	

#### Predecessor

None

#### Responsible

None

#### Inputs

None

#### Guidance

None

#### Successor

None

#### Outputs

None

#### Phases

None

#### 1.7.5.1 Review phasing for flight software (5.3.6.1)

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

1. the system model philosophy (e.g. protoflight model, versus utilization of engineering qualification model)
2. the system verification and qualification approach and constraints
3. the capability to baseline the system design at system CDR, by knowing enough information about software design, in particular consolidated sizing and timing budgets, consistent hardware design and software design.

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

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

#### Predecessor

None

#### Responsible

None

#### Supporting

None

#### Successor

None

#### Accountable

None

#### Consulted

None

<b>Informed</b>
None
<b>Inputs</b>
None
<b>Guidance</b>
None

<b>Outputs</b>
<ul style="list-style-type: none"><li>Flight software review phasing</li></ul>
<b>Phases</b>
None

1.7.5.2 Review phasing for ground software (5.3.6.2)

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

- 1. the baseline of the software requirements (e.g. SWRR) is performed before the ground segment PDR ,
- 2. the software PDR is performed before the ground segment CDR
- 3. all the other software reviews are performed before the ground segment QR.

<b>Predecessor</b>
None
<b>Responsible</b>
None
<b>Supporting</b>
None
<b>Informed</b>
None
<b>Inputs</b>
None
<b>Guidance</b>
None

<b>Successor</b>
None
<b>Accountable</b>
None
<b>Consulted</b>
None
<b>Outputs</b>
<ul style="list-style-type: none"><li>Ground software review phasing</li></ul>
<b>Phases</b>
None

1.7.6 Interface management (5.3.7)

Supplier	Inputs	Workflow	Outputs	Customer
		<div>Interface management procedures (5.3.7.1)</div>	<div> Interface management procedures</div>	

<b>Predecessor</b>
None
<b>Responsible</b>
None

<b>Successor</b>
None

<b>Inputs</b>
None
<b>Guidance</b>
None

<b>Outputs</b>
None
<b>Phases</b>
None



1.7.6.1 Interface management procedures (5.3.7.1)

Interface management procedures shall be defined in accordance with ECSSMST40 requirements.

<b>Predecessor</b>
None
<b>Responsible</b>
None
<b>Supporting</b>
None
<b>Informed</b>
None
<b>Inputs</b>
None
<b>Guidance</b>
None

<b>Successor</b>
None
<b>Accountable</b>
None
<b>Consulted</b>
None
<b>Outputs</b>
<ul style="list-style-type: none"><li>Interface management procedures</li></ul>
<b>Phases</b>
None

1.7.7 Technical budget and margin management (5.3.8)

Supplier	Inputs	Workflow	Outputs	Customer
		Software technical budget and margin philosophy definition (5.3.8.1)	 Technical budgets and margin philosophy for the project	
		Technical budget and margin computation (5.3.8.2)	 Technical budgets and margin computation	

<b>Predecessor</b>
None
<b>Responsible</b>
None
<b>Inputs</b>
None

<b>Successor</b>
None
<b>Outputs</b>
None

**Guidance**

None

**Phases**

None

**1.7.7.1 Software technical budget and margin philosophy definition (5.3.8.1)**

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

**Predecessor**

None

**Successor**

None

**Responsible**

None

**Accountable**

None

**Supporting**

None

**Consulted**

None

**Informed**

None

**Inputs**

None

**Outputs**

- Technical budgets and margin philosophy for the project

**Guidance**

None

**Phases**

None

**1.7.7.2 Technical budget and margin computation (5.3.8.2)**

The way to compute the technical budgets and margin shall be agreed between the customer and the supplier.

**Predecessor**

None

**Successor**

None

**Responsible**

None

**Accountable**

None

**Supporting**

None

**Consulted**

None

**Informed**

None

**Inputs**

None

**Outputs**

- Technical budgets and margin computation

**Guidance**



None

**Phases**

None



1.7.8 Compliance to this Standard (5.3.9)

Supplier	Inputs	Workflow	Outputs	Customer
		Compliance matrix (5.3.9.1)	 ECSS-E-ST-40 compliance matrix	
		Documentation compliance (5.3.9.2)	 ECSS-E-ST-40 compliance matrix	

<b>Predecessor</b>
None
<b>Responsible</b>
None
<b>Inputs</b>
None
<b>Guidance</b>
None

<b>Successor</b>
None
<b>Outputs</b>
None
<b>Phases</b>
None

1.7.8.1 Compliance matrix (5.3.9.1)

The supplier shall provide a compliance matrix documenting conformance with the individual software engineering process requirements (Clause 5) applicable to the project or business agreement, as per ECSSST00.

<b>Predecessor</b>
None
<b>Responsible</b>
None
<b>Supporting</b>
None
<b>Informed</b>
None
<b>Inputs</b>
None
<b>Guidance</b>
None

<b>Successor</b>
None
<b>Accountable</b>
None
<b>Consulted</b>
None
<b>Outputs</b>
<ul style="list-style-type: none"><li>ECSEST40 compliance matrix</li></ul>
<b>Phases</b>
None

1.7.8.2 Documentation compliance (5.3.9.2)

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"><li>ECSSEST40 compliance matrix</li></ul>
<b>Guidance</b>	<b>Phases</b>
None	None

## 1.8 Software maintenance process (5.10)

Supplier	Inputs	Workflow	Outputs	Customer
		<div>Process implementation (5.10.2) +</div>		
		<div>Problem and modification analysis (5.10.3) +</div>		
		<div>Modification implementation (5.10.4) +</div>		
		<div>Conducting maintenance reviews (5.10.5) +</div>		
		<div>Software migration (5.10.6) +</div>		
		<div>Software retirement (5.10.7) +</div>		

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

The maintenance process contains the activities and tasks of the maintainer. The objective is to modify an existing software product while preserving its integrity. This process includes the migration and retirement of the software product. The process ends with the retirement of the software product. The activities provided in clause 5.10 are specific to the maintenance process; however, the process can utilize other processes in this Standard. If the software engineering processes are utilized, the term supplier there is interpreted as maintainer.

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

### Predecessor

- Software management process (5.3)

### Successor

- Software operation process (5.9)

### Responsible

- Customer

### Inputs

None

### Outputs

None






**Guidance**

None

**Phases**

None

**1.8.1 Process implementation (5.10.2)**

Supplier	Inputs	Workflow	Outputs	Customer
		<div>Establishment of the software maintenance process (5.10.2.1)</div>	<div>  Maintenance plan ☒ plans and procedures            Maintenance plan ☒ applicability of development process              procedures, methods, tools and standards            Maintenance plan ☒ configuration management process              problem reporting and handling              Problem and nonconformance report         </div>	
		<div>Long term maintenance for flight software (5.10.2.2)</div>	<div>  Maintenance plan ☒ long term maintenance solutions         </div>	

**Predecessor**

None

**Successor**

None

**Responsible**

None

**Inputs**

None

**Outputs**

None

**Guidance**

None

**Phases**

None

**1.8.1.1 Establishment of the software maintenance process (5.10.2.1)**

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

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

The maintainer shall implement (or establish the organizational interface with) the configuration management process (ECSSMST40) for managing modifications.

The maintainer shall establish procedures for receiving, recording and tracking problem reports and modification

requests, providing feedback to the requester.

Whenever problems are encountered, they shall be recorded and entered in accordance with the change control established and maintained in conformance with ECSSMST40.

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>• Maintenance plan plans and procedures</li> <li>• Maintenance plan applicability of development process procedures, methods, tools and standards</li> <li>• Maintenance plan configuration management process</li> <li>• Maintenance plan problem reporting and handling</li> <li>• Problem and nonconformance report</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

#### 1.8.1.2 Long term maintenance for flight software (5.10.2.2)

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>• Maintenance plan long term maintenance solutions</li> </ul>
<b>Guidance</b>	<b>Phases</b>

None

None

1.8.2 Problem and modification analysis (5.10.3)

Supplier	Inputs	Workflow	Outputs	Customer
		<div>Problem analysis (5.10.3.1)</div>	<div><div></div>Modification analysis report</div> <div><div></div>Modification approval</div> <div><div></div>Problem analysis report</div>	

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	None
<b>Guidance</b>	<b>Phases</b>
None	None

1.8.2.1 Problem analysis (5.10.3.1)

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

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

The maintainer shall reproduce or verify the problem.

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

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

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None

**Informed**

None

**Inputs**

None

**Outputs**

- Modification analysis report
- Modification approval
- Problem analysis report




**Guidance**

None

**Phases**

None

**1.8.3 Modification implementation (5.10.4)**

Supplier	Inputs	Workflow	Outputs	Customer
		Analysis and documentation of product modification (5.10.4.1)	 Modification documentation	
		Documentation of software product changes (5.10.4.2)	 Modification documentation	
		Invoking of software engineering processes for modification...	 Modification documentation	

**Predecessor**

None

**Successor**

None

**Responsible**

None

**Inputs**

None

**Outputs**

None

**Guidance**

None

**Phases**

None

**1.8.3.1 Analysis and documentation of product modification (5.10.4.1)**

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

**Predecessor**

None

**Successor**

None

**Responsible****Accountable**

None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>Modification documentation</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

### 1.8.3.2 Documentation of software product changes (5.10.4.2)

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>Modification documentation</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

### 1.8.3.3 Invoking of software engineering processes for modification implementation (5.10.4.3)

The maintainer shall apply the software engineering processes as specified in clauses 5.3 to 5.8 while implementing the modifications

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

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



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

The test results shall be documented.



<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"><li>Modification documentation</li></ul>
<b>Guidance</b>	<b>Phases</b>
None	None

1.8.4 Conducting maintenance reviews (5.10.5)

Supplier	Inputs	Workflow	Outputs	Customer
		Maintenance reviews (5.10.5.1)	 MF - Joint review reports	
		Baseline for change (5.10.5.2)	 Baseline for change	

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	None
<b>Guidance</b>	<b>Phases</b>
None	None

1.8.4.1 Maintenance reviews (5.10.5.1)

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

<b>Predecessor</b>	<b>Successor</b>
None	None

<b>Responsible</b>
None
<b>Supporting</b>
None
<b>Informed</b>
None
<b>Inputs</b>
None
<b>Guidance</b>
None

<b>Accountable</b>
None
<b>Consulted</b>
None
<b>Outputs</b>
<ul style="list-style-type: none"><li>MF - Joint review reports</li></ul>
<b>Phases</b>
None

1.8.4.2 Baseline for change (5.10.5.2)

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

<b>Predecessor</b>
None
<b>Responsible</b>
None
<b>Supporting</b>
None
<b>Informed</b>
None
<b>Inputs</b>
None
<b>Guidance</b>
None

<b>Successor</b>
None
<b>Accountable</b>
None
<b>Consulted</b>
None
<b>Outputs</b>
<ul style="list-style-type: none"><li>Baseline for change</li></ul>
<b>Phases</b>
None

**1.8.5 Software migration (5.10.6)**

Supplier	Inputs	Workflow	Outputs	Customer
		Applicability of this Standard to software migration (5.10.6.1)	 Migration plan and notification	
		Migration planning and execution (5.10.6.2)	 Migration plan and notification	
		Contribution to the migration plan (5.10.6.3)	 Migration plan and notification	
		Preparation for migration (5.10.6.4)	 Migration plan and notification	
		Notification of transition to migrated system (5.10.6.5)	 Migration plan and notification	
		Post-operation review (5.10.6.6)	 Post operation review report	
		Maintenance and accessibility of data of former system (5.10.6.7)	 Migration plan and notification	

**Predecessor**

None

**Responsible**

None

**Inputs**

None

**Guidance**

None

**Successor**

None

**Outputs**

None

**Phases**

None

### 1.8.5.1 Applicability of this Standard to software migration (5.10.6.1)

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

#### Predecessor

None

#### Responsible

None

#### Supporting

None

#### Informed

None

#### Inputs

None

#### Guidance

None

#### Successor

None

#### Accountable

None

#### Consulted

None

#### Outputs

- Migration plan and notification

#### Phases

None

### 1.8.5.2 Migration planning and execution (5.10.6.2)

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

1. requirements analysis and definition of migration;
2. development of migration tools;
3. conversion of software product and data;
4. migration execution;
5. migration verification;
6. support for the old environment in the future;
7. operator involvement in the activities.

#### Predecessor

None

#### Responsible

None

#### Supporting

None

#### Informed

None

#### Inputs

None

#### Guidance

None

#### Successor

None

#### Accountable

None

#### Consulted

None

#### Outputs

- Migration plan and notification

#### Phases

None

### 1.8.5.3 Contribution to the migration plan (5.10.6.3)

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

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

#### Predecessor

None

#### Responsible

None

#### Supporting

None

#### Informed

None

#### Inputs

None

#### Guidance

None

#### Successor

None

#### Accountable

None

#### Consulted

None

#### Outputs

- Migration plan and notification

#### Phases

None

### 1.8.5.4 Preparation for migration (5.10.6.4)

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

#### Predecessor

None

#### Responsible

None

#### Supporting

None

#### Informed

None

#### Inputs

None

#### Guidance

None

#### Successor

None

#### Accountable

None

#### Consulted

None

#### Outputs

- Migration plan and notification

#### Phases

None

**1.8.5.5 Notification of transition to migrated system (5.10.6.5)**

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

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>• Migration plan and notification</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

**1.8.5.6 Post-operation review (5.10.6.6)**

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

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





<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>• Post operation review report</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

1.8.5.7 Maintenance and accessibility of data of former system (5.10.6.7)

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"><li>Migration plan and notification</li></ul>
<b>Guidance</b>	<b>Phases</b>
None	None

1.8.6 Software retirement (5.10.7)

Supplier	Inputs	Workflow	Outputs	Customer
		Retirement planning (5.10.7.1)	 Retirement plan and notification	
		Notification of retirement plan (5.10.7.2)	 Retirement plan and notification	
		Identification of requirements for software retirement (5.10.7.3)	 Retirement plan and notification	
		Maintenance and accessibility to data of the retired product (5.10.7.4)	 Retirement plan and notification	

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	

None	
<b>Inputs</b>	<b>Outputs</b>
None	None
<b>Guidance</b>	<b>Phases</b>
None	None

#### 1.8.6.1 Retirement planning (5.10.7.1)

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

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>• Retirement plan and notification</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

#### 1.8.6.2 Notification of retirement plan (5.10.7.2)

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

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None



**Supporting**

None

**Informed**

None

**Inputs**

None

**Guidance**

None

**Consulted**

None

**Outputs**

- Retirement plan and notification

**Phases**

None

**1.8.6.3 Identification of requirements for software retirement (5.10.7.3)**

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

**Predecessor**

None

**Responsible**

None

**Supporting**

None

**Informed**

None

**Inputs**

None

**Guidance**

None

**Successor**

None

**Accountable**

None

**Consulted**

None

**Outputs**

- Retirement plan and notification

**Phases**

None

**1.8.6.4 Maintenance and accessibility to data of the retired product (5.10.7.4)**

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

**Predecessor**

None

**Responsible**

None

**Supporting**

None

**Informed**

None

**Successor**

None

**Accountable**

None

**Consulted**

None

**Inputs**

None

**Outputs**

- Retirement plan and notification

**Guidance**

None

**Phases**

None

**1.9 Software operation process (5.9)**

Supplier	Inputs	Workflow	Outputs	Customer
		<div>Process implementation (5.9.2) +</div>		
		<div>Operational testing (5.9.3) +</div>		
		<div>Software operation support (5.9.4) +</div>		
		<div>User support (5.9.5) +</div>		

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

Indeed, software operation in this Standard is totally different from spacecraft or payload operations performed by ground stations on space systems.

Software operation in this Standard includes mainly the helpdesk and the link between the users, the developers or maintainers, and the customer.

**Predecessor**

- Software maintenance process (5.10)

**Successor**

None

**Responsible**

- Customer

**Inputs**

None

**Outputs**

None

**Guidance**

None

**Phases**

None

**1.9.1 Process implementation (5.9.2)**

Supplier	Inputs	Workflow	Outputs	Customer
		Operational testing definition (5.9.2.1)	Software operation support plan ☒ operational testing specifications	
		Software operation support plans and procedures development (5.9.2.2)	Software operation support plan ☒ plans and procedures	
		Problem handling procedures definition (5.9.2.3)	Software operation support plan ☒ procedures for problem handling	

**Predecessor**

None

**Responsible**

None

**Inputs**

None

**Guidance**

None

**Successor**

None

**Outputs**

None

**Phases**

None

**1.9.1.1 Operational testing definition (5.9.2.1)**

The SOS entity shall establish procedures for:

1. testing the software product in its operational environment;
2. entering problem reports and modification requests to the maintenance process (see clause 5.10), and;
3. releasing the software product for operational use in accordance with the change control established and maintained in conformance with ECSSMST40.

**Predecessor**

None

**Responsible**

None

**Supporting**

None

**Informed**

None

**Successor**

None

**Accountable**

None

**Consulted**

None

**Inputs**

None

**Outputs**

- Software operation support plan operational testing specifications

**Guidance**

None

**Phases**

None

**1.9.1.2 Software operation support plans and procedures development (5.9.2.2)**

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

**Predecessor**

None

**Successor**

None

**Responsible**

None

**Accountable**

None

**Supporting**

None

**Consulted**

None

**Informed**

None

**Inputs**

None

**Outputs**

- Software operation support plan plans and procedures

**Guidance**

None

**Phases**

None

**1.9.1.3 Problem handling procedures definition (5.9.2.3)**

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

**Predecessor**

None

**Successor**

None

**Responsible**

None

**Accountable**

None

**Supporting**

None

**Consulted**

None

**Informed**

None

**Inputs**

None

**Outputs**

- Software operation support plan procedures for problem handling

<b>Guidance</b>	<b>Phases</b>
None	None

1.9.2 Operational testing (5.9.3)

Supplier	Inputs	Workflow	Outputs	Customer
		Operational testing execution (5.9.3.1)	 Operational testing results	
		Software operational requirements demonstration (5.9.3.2)	 Operational testing results	
		Software release (5.9.3.3)	 Software product and media labels	

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	None
<b>Guidance</b>	<b>Phases</b>
None	None

1.9.2.1 Operational testing execution (5.9.3.1)

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>

None	<ul style="list-style-type: none"> <li>Operational testing results</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

### 1.9.2.2 Software operational requirements demonstration (5.9.3.2)

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

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

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

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	
<b>Inputs</b>	<b>Outputs</b>
None	<ul style="list-style-type: none"> <li>Operational testing results</li> </ul>
<b>Guidance</b>	<b>Phases</b>
None	None

### 1.9.2.3 Software release (5.9.3.3)

The software product shall be released for operational use.

<b>Predecessor</b>	<b>Successor</b>
None	None
<b>Responsible</b>	<b>Accountable</b>
None	None
<b>Supporting</b>	<b>Consulted</b>
None	None
<b>Informed</b>	
None	

**Inputs**

None

**Outputs**

- Software product and media labels


**Guidance**

None

**Phases**

None

**1.9.3 Software operation support (5.9.4)**

Supplier	Inputs	Workflow	Outputs	Customer
		Software operation support performance (5.9.4.1)		
		Problem handling (5.9.4.2)	 Problem and nonconformance report	

**Predecessor**

None

**Successor**

None

**Responsible**

None

**Inputs**

None

**Outputs**

None

**Guidance**

None

**Phases**

None

**1.9.3.1 Software operation support performance (5.9.4.1)**

The software operation support plan shall be executed.

**Predecessor**

None

**Successor**

None

**Responsible**

None

**Accountable**

None

**Supporting**

None

**Consulted**

None

**Informed**

None

**Inputs**

None

**Outputs**

None

Guidance

None

Phases

None

1.9.3.2 Problem handling (5.9.4.2)

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

Predecessor

None

Successor

None

Responsible

None

Accountable

None

Supporting

None

Consulted

None

Informed

None

Inputs

None

Outputs

- Problem and nonconformance report




Guidance

None

Phases

None

1.9.4 User support (5.9.5)

Supplier	Inputs	Workflow	Outputs	Customer
		Assistance to the user (5.9.5.1)	User's request record  user's request and subsequent actions	
		Handling of user's requests (5.9.5.2)	User's request record  actions	
		Provisions of work-around solutions (5.9.5.3)	User's request record  work around solution	

Predecessor

None

Successor

None

Responsible

None

Inputs

Outputs



None
<b>Guidance</b>
None

None
<b>Phases</b>
None

#### 1.9.4.1 Assistance to the user (5.9.5.1)

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

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

<b>Predecessor</b>
None
<b>Responsible</b>
None
<b>Supporting</b>
None
<b>Informed</b>
None
<b>Inputs</b>
None
<b>Guidance</b>
None

<b>Successor</b>
None
<b>Accountable</b>
None
<b>Consulted</b>
None
<b>Outputs</b>
<ul style="list-style-type: none"> <li>User's request record user's request and subsequent actions</li> </ul>
<b>Phases</b>
None

#### 1.9.4.2 Handling of user's requests (5.9.5.2)

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

The SOS entity shall address user's requests.

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

<b>Predecessor</b>
None
<b>Responsible</b>
None
<b>Supporting</b>
None
<b>Informed</b>
None
<b>Inputs</b>
None

<b>Successor</b>
None
<b>Accountable</b>
None
<b>Consulted</b>
None
<b>Outputs</b>
<ul style="list-style-type: none"> <li>User's request record actions</li> </ul>

Guidance
None

Phases
None

1.9.4.3 Provisions of work-around solutions (5.9.5.3)

If a reported problem has a temporary work-around solution before a permanent solution can be released, the SOS entity shall give to the originator of the problem report the option to use it.  
Permanent corrections, releases that include previously omitted functions or features, and system improvements shall be applied to the operational software product using the maintenance process as specified in clause 5.10.

Predecessor
None

Successor
None

Responsible
None

Accountable
None

Supporting
None

Consulted
None

Informed
None

Inputs
None

Outputs
<ul style="list-style-type: none"><li>User’s request record work around solution</li></ul>

Guidance
None

Phases
None