



EULYNX Initiative

EULYNX Partners:

Bane NOR
Société Nationale des Chemins de Fer Luxembourgeois (CFL)
DB Netz AG (DB)
S.A. Infrabel
Liikennevirasto (FTA)
Network Rail
ProRail B.V.
Rete Ferroviaria Italiana (RFI)
SBB AG
Société Nationale des Chemins de Fer Français (SNCF)
SŽ-Infrastruktura, d.o.o. (SŽ)
Trafikverket

System engineering process

Document number: Eu.Doc.27

Baseline: 2.0 (0.A)

EULYNX Baseline Set: 2



Inhalt

1	Introduction	1
1.1	Release information	1
1.2	Impressum	2
1.3	Purpose	3
1.4	Applicable standards and regulations	3
1.5	Applicable documents	4
1.6	Terms and abbreviations	4
1.7	Definition of object types	4
2	Concept	4
2.1	Relation to EN 50126	4
2.2	Relation to model based systems engineering	4
2.3	Iterative-incremental process model	4
3	Artefacts	6
3.1	Functional list	6
3.2	Model	7
3.3	Model subset	7
3.4	Functional specification	7
3.5	Interface definition	7
3.6	Interface specification	7
3.7	System definition	7
3.8	Conceptual document	7
3.9	Model test specification	8
3.10	System test specification	8
3.11	Validation checklist	8
4	Stakeholders/Roles	9

4.1	Cluster leader	9
4.2	Cluster engineer	9
4.3	Model verifier	9
4.4	Model validator	9
4.5	System tester	9
4.6	Specification validator	9
5	Activities	10
5.1	Edit conceptual documents	10
5.2	Edit functional list	10
5.3	MBSE activities	10
5.3.1	MBSE - subsystem definition	11
5.3.2	MBSE - logical architecture	12
5.3.3	MBSE - model verification	12
5.3.4	MBSE - model validation	12
5.4	Edit specification documents	13
5.5	Validate specification documents	14
5.6	Edit system test specification	14
5.7	Perform acceptance tests	15
5.8	Management activities	15
5.8.1	Configuration management (CM)	15
5.8.1.1	Variability management	15
5.8.1.2	Model maintenance	16
5.8.1.3	Baseline management	16
5.8.2	Quality management (QM)	16
5.8.2.1	General QM tasks	16
5.8.2.2	Specific QM tasks	16
5.8.3	Safety management (SM)	17
5.8.4	Change control management (CCM)	17
6	Phases	17

6.1	Concept phase	17
6.2	Development phase	18
6.2.1	SubSysDef - Subsystem definition	18
6.2.2	LogArch - logical architecture and validation	19
6.2.3	TestGen - generation of system test specification	20
6.3	System acceptance phase	21
6.4	Maintenance phase	22
7	Process instances	23

ID	Type	Requirements
Eu.SEP.1	Head	1 Introduction
Eu.SEP.6	Head	1.1 Release information
Eu.SEP.7	Info	[Eu.Doc.27] System engineering process CENELEC Phase: 1-5 Version: 2.0 (0.A) EULYNX Baseline Set: 2 Approval date: 30.11.2017
Eu.SEP.5	Info	Version history
Eu.SEP.8	Info	version number: 0.0 date: 23.11.2016 author: Oliver Lemke review: - changes: initial version, outline only
Eu.SEP.9	Info	version number: 0.1 (0.A) date: 02.02.2017 author: Oliver Lemke review: - changes: version for SIGNON internal review
Eu.SEP.10	Info	version number: 0.1 (1.A) date: 14.03.2017 author: Oliver Lemke review: Marie Killat changes: version for cluster review
Eu.SEP.11	Info	version number: 0.1 (2.A) date: 10.05.2017 author: Oliver Lemke review: Mirko Blazic changes: amended by comments of Mirko Blazic
Eu.SEP.35	Info	version number: 0.1 (3.A) date: 31.05.2017 author: Marie Killat review: Dennis Kunz, Maria Bertram changes: transfer from Word to DOORS for release, update of imprint, internal quality review

ID	Type	Requirements
Eu.SEP.220	Info	version number: 1.0 (0.A) date: 26.06.2017 author: Dennis Kunz review: CCB changes: EUMT-20
Eu.SEP.222	Info	version number: 1.1 (0.A) date: 26.10.2017 author: Charlotte Gäbel review: - changes: EUMT-33
Eu.SEP.228	Info	version number: 1.1 (1.A) date: 27.10.2017 author: Charlotte Gäbel review: Darren Witts changes: improved terminology
Eu.SEP.229	Info	version number: 2.0 (0.A) date: 05.12.2017 author: Charlotte Gäbel review: CCB changes: EUAR-147, EUAR-148, EUAR-149, EUAR-150, EUAR-163
Eu.SEP.12	Head	1.2 Impressum

ID	Type	Requirements
Eu.SEP.13	Info	<p>Publisher: EULYNX Initiative</p> <p>EULYNX Partners: Bane NOR Société Nationale des Chemins de Fer Luxembourgeois (CFL) DB Netz AG (DB) S.A. Infrabel Liikennevirasto (FTA) Network Rail ProRail B.V. Rete Ferroviaria Italiana (RFI) SBB AG Société Nationale des Chemins de Fer Français (SNCF) SZ-Infrastruktura, d.o.o. (SZ) Trafikverket</p>
Eu.SEP.14	Info	<p>Responsible for this document: EULYNX Project Management Office www.eulynx.eu</p>
Eu.SEP.15	Info	<p>Copyright EULYNX Partners All information included or disclosed in this document is licensed under the European Union Public Licence EUPL, Version 1.1.</p>
Eu.SEP.16	Head	1.3 Purpose
Eu.SEP.17	Info	This document describes the systems engineering process for the EULYNX project on a high level of abstraction. It is intended for stakeholders involved in the modelling and specification work to get an overview over the basic systems engineering methodology within the project scope.
Eu.SEP.18	Info	This document does not contain detailed technical instructions or the usage of tools. Instead, it will reference to other documents for that purpose.
Eu.SEP.19	Info	<p>This document is intended for the following users:</p> <ul style="list-style-type: none"> • safety authorities • infrastructure managers • safety assessors • signalling system suppliers • validators
Eu.SEP.20	Head	1.4 Applicable standards and regulations
Eu.SEP.21	Info	A list of applicable standards and regulations used in EULYNX is listed in the EULYNX Reference Document List [Eu.Doc.12].

ID	Type	Requirements
Eu.SEP.22	Head	1.5 Applicable documents
Eu.SEP.23	Info	The current versions of documents used as input or related to this document are listed in the EULYNX Documentation Plan [Eu.Doc.11]. The relationships between the documents are displayed in the Appendix A1 Documentation plan and structure [Eu.Doc.11_A1].
Eu.SEP.32	Head	1.6 Terms and abbreviations
Eu.SEP.33	Info	The terms and abbreviations are listed in the EULYNX Glossary [Eu.Doc.9].
Eu.SEP.213	Head	1.7 Definition of object types
Eu.SEP.219	Info	The following definition for object types is applied in this document:
Eu.SEP.214	Info	<ul style="list-style-type: none"> • "Info" - This denotes additional information to help understand the specification. These objects do not specify any additional requirements.
Eu.SEP.218	Info	<ul style="list-style-type: none"> • "Head" - This denotes chapter headings.
Eu.SEP.3	Head	2 Concept
Eu.SEP.34	Head	2.1 Relation to EN 50126
Eu.SEP.4	Info	<p>The EULYNX systems engineering process is closely oriented on the life cycle phases defined in EN 50126 and covers the following phases:</p> <ul style="list-style-type: none"> • 1 - concept • 2 - system definition • 4 - system requirements • 5 - apportionment of system requirements • 10 - system acceptance • 11 - operation and maintenance <p>See Fig. 1 - Process model for association of CENELEC phases to phases of this process.</p>
Eu.SEP.36	Head	2.2 Relation to model based systems engineering
Eu.SEP.2	Info	<i>Note: intentionally left blank. To be filled in later version of this document.</i>
Eu.SEP.37	Head	2.3 Iterative-incremental process model

ID	Type	Requirements
Eu.SEP.38	Info	The core element of the SE process is an iterative-incremental process model. These process models slice the overall task into a number of iterations. Every iteration then processes a subset of the overall task by executing a set of activities. The results of an iteration are appended to the existing artefacts of prior iterations, hence increasing the complexity of the artefacts over time.
Eu.SEP.39	Info	For EULYNX, a modified setup is used: iterations themselves are grouped into phases, where every phase defines a different set of activities for the iterations. As a basic rule, iterations of early phases usually contain fewer activities than iterations of later phases. Therefore, as the project progresses, the iterations become more and more complex and comprise more activities.
Eu.SEP.40	Info	<p>Fig. 1 shows a graphical, table-like representation of this process model. Activities are displayed as rows, phases and iterations as columns. For every cell, a colour or a text indicate whether an activity is to be executed or not in the particular iteration:</p> <ul style="list-style-type: none"> • A white cell indicates an activity not to be performed within the iteration • A white cell with text “CCM” indicates that the activity needs only to be performed if change control management agreed upon changes • A dark cell indicates an activity to be performed within the iteration <p>Every iteration processes all “dark cell” activities in the column.</p>
Eu.SEP.41	Info	Fig. 1 – EULYNX Process model

ID	Type	Requirements																																																																																																																																																																																																																																																
Eu.SEP.41		<table><tr><td>Activities</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>Edit conceptual docs</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>CCM</td><td>CCM</td><td>CCM</td><td>CCM</td><td>CCM</td></tr><tr><td>Edit functional lists</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>CCM</td><td>CCM</td><td>CCM</td><td>CCM</td><td>CCM</td></tr><tr><td>MBSE - subsystem definition</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>CCM</td><td>CCM</td><td>CCM</td><td>CCM</td><td>CCM</td></tr><tr><td>MBSE - logical architecture</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>CCM</td><td>CCM</td><td>CCM</td><td>CCM</td><td>CCM</td></tr><tr><td>MBSE - model validation</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>CCM</td><td>CCM</td><td>CCM</td><td>CCM</td><td>CCM</td></tr><tr><td>Edit spec. documents</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>CCM</td><td>CCM</td><td>CCM</td><td>CCM</td><td>CCM</td></tr><tr><td>Validate spec. documents</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>CCM</td><td>CCM</td><td>CCM</td><td>CCM</td><td>CCM</td></tr><tr><td>Edit system test specification</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>CCM</td><td>CCM</td><td>CCM</td><td>CCM</td><td>CCM</td></tr><tr><td>Perform acceptance tests</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>CCM</td><td>CCM</td><td>CCM</td></tr><tr><td>CM, QM, SM, CCM</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>Iteration</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td><td></td></tr><tr><td>Sub phases</td><td></td><td></td><td colspan="2">SubSysDef</td><td colspan="3">LogArch</td><td colspan="3">TestGen</td><td colspan="5"></td></tr><tr><td>Phases</td><td colspan="2">Concept</td><td colspan="8">Development</td><td colspan="2">System test</td><td colspan="2">Maintenance</td><td></td></tr><tr><td>Associated CENELEC phase</td><td colspan="2">1-2</td><td colspan="8">4-5</td><td colspan="2">10</td><td colspan="2">11</td><td></td></tr></table> <div><div></div><div>time</div></div>	Activities																Edit conceptual docs											CCM	CCM	CCM	CCM	CCM	Edit functional lists											CCM	CCM	CCM	CCM	CCM	MBSE - subsystem definition											CCM	CCM	CCM	CCM	CCM	MBSE - logical architecture											CCM	CCM	CCM	CCM	CCM	MBSE - model validation											CCM	CCM	CCM	CCM	CCM	Edit spec. documents											CCM	CCM	CCM	CCM	CCM	Validate spec. documents											CCM	CCM	CCM	CCM	CCM	Edit system test specification											CCM	CCM	CCM	CCM	CCM	Perform acceptance tests													CCM	CCM	CCM	CM, QM, SM, CCM																Iteration	1	2	3	4	5	6	7	8	9	10	11	12	13	14		Sub phases			SubSysDef		LogArch			TestGen								Phases	Concept		Development								System test		Maintenance			Associated CENELEC phase	1-2		4-5								10		11		
		Activities																																																																																																																																																																																																																																																
		Edit conceptual docs											CCM	CCM	CCM	CCM	CCM																																																																																																																																																																																																																																	
		Edit functional lists											CCM	CCM	CCM	CCM	CCM																																																																																																																																																																																																																																	
		MBSE - subsystem definition											CCM	CCM	CCM	CCM	CCM																																																																																																																																																																																																																																	
		MBSE - logical architecture											CCM	CCM	CCM	CCM	CCM																																																																																																																																																																																																																																	
		MBSE - model validation											CCM	CCM	CCM	CCM	CCM																																																																																																																																																																																																																																	
		Edit spec. documents											CCM	CCM	CCM	CCM	CCM																																																																																																																																																																																																																																	
		Validate spec. documents											CCM	CCM	CCM	CCM	CCM																																																																																																																																																																																																																																	
		Edit system test specification											CCM	CCM	CCM	CCM	CCM																																																																																																																																																																																																																																	
		Perform acceptance tests													CCM	CCM	CCM																																																																																																																																																																																																																																	
		CM, QM, SM, CCM																																																																																																																																																																																																																																																
		Iteration	1	2	3	4	5	6	7	8	9	10	11	12	13	14																																																																																																																																																																																																																																		
		Sub phases			SubSysDef		LogArch			TestGen																																																																																																																																																																																																																																								
		Phases	Concept		Development								System test		Maintenance																																																																																																																																																																																																																																			
Associated CENELEC phase	1-2		4-5								10		11																																																																																																																																																																																																																																					
Eu.SEP.42	Head	3 Artefacts																																																																																																																																																																																																																																																
Eu.SEP.43	Info	This chapter defines the artefacts created during the application of this systems engineering process.																																																																																																																																																																																																																																																
Eu.SEP.44	Head	3.1 Functional list																																																																																																																																																																																																																																																
Eu.SEP.45	Info	A FUNCTIONAL LIST is an informal collection of functionality for a given subsystem or interface. Its intended usage is to provide a tool for early functional analysis within EULYNX clusters. A FUNCTIONAL LIST also contains attributes to indicate for which IMs a function is applicable. There is no unified EULYNX requirement about the structure, contents or format of a FUNCTIONAL LIST. This is left to the cluster leaders.																																																																																																																																																																																																																																																

ID	Type	Requirements
Eu.SEP.46	Head	3.2 Model
Eu.SEP.47	Info	The MODEL refers to the EULYNX SysML model in its entirety. It comprises all model elements of all interfaces and all subsystems.
Eu.SEP.48	Head	3.3 Model subset
Eu.SEP.49	Info	The MODEL SUBSET refers to a subset of the MODEL. This subset might be defined by technical constructs (like a SysML-package containing all model elements), formal definitions or informal agreements between stakeholders. E.g. a MODEL SUBSET could comprise all model elements necessary to describe a particular subsystem. There is no formal EULYNX requirement on how a MODEL SUBSET for a particular usage is to be defined. This document however makes use of informal definitions of model subsets.
Eu.SEP.50	Head	3.4 Functional specification
Eu.SEP.51	Info	<p>A FUNCTIONAL SPECIFICATION consists of a number of requirements specifying the functional and non-functional aspects of a EULYNX subsystem. These requirements are contained within in a formal DOORS module, adhering to a predefined template structure. Also the export of the DOORS module in any different format (PDF, Word, etc.) might be referred to as a FUNCTIONAL SPECIFICATION.</p> <p>For every FUNCTIONAL SPECIFICATION there is a MODEL SUBSET comprising the model elements specifying the functional aspects of a subsystem. Parts of the FUNCTIONAL SPECIFICATION can therefore be generated automatically out of the MODEL SUBSET.</p>
Eu.SEP.52	Head	3.5 Interface definition
Eu.SEP.53	Info	<p>An INTERFACE DEFINITION consists of a number of requirements specifying the protocol stack of a EULYNX interface. These comprise the definition of the interface layers according to the ISO-OSI model and the specification of configuration parameters if necessary. These requirements are contained within in a formal DOORS module, adhering to a predefined template structure. Also the export of the DOORS module in any different format (PDF, Word, etc.) might be referred to as an INTERFACE DEFINITION.</p> <p>INTERFACE DEFINITIONS are usually created manually within DOORS.</p>
Eu.SEP.54	Head	3.6 Interface specification
Eu.SEP.55	Info	<p>An INTERFACE SPECIFICATION consists of a number of requirements specifying the telegram structure of a EULYNX interface. These comprise the definition of the telegrams themselves as well as the definition of allowed values for telegram parameters. These requirements are contained within in a formal DOORS module, adhering to a predefined template structure. Also the export of the DOORS module in any different format (PDF, Word, etc.) might be referred to as an INTERFACE SPECIFICATION.</p> <p>INTERFACE SPECIFICATIONS are usually created manually within DOORS.</p>
Eu.SEP.56	Head	3.7 System definition
Eu.SEP.57	Info	The SYSTEM DEFINITION [Eu.Doc.7] consists of a document describing the system structure of the EULYNX architecture and a diagram showing the system reference architecture graphically. It defines the subsystems, the systems and actors in the system environment and the interfaces connecting them.
Eu.SEP.58	Head	3.8 Conceptual document

ID	Type	Requirements
Eu.SEP.59	Info	A CONCEPTUAL DOCUMENT is a textual document, an informal graphic or any other document describing an aspect of the EULYNX project besides the specification documents themselves. It usually covers early phases of the development process and describes abstract concepts. There is no formal requirement for format or structure of a CONCEPTUAL DOCUMENT.
Eu.SEP.60	Head	3.9 Model test specification
Eu.SEP.61	Info	<p>A MODEL TEST SPECIFICATION is a collection of test cases or test steps for a given MODEL SUBSET. These test cases are intended for the validation of a given MODEL SUBSET. There are no strict requirements defined for the structure or format of a MODEL TEST SPECIFICATION in this document, but two basic types of test cases for a MODEL TEST SPECIFICATION can be distinguished:</p> <ul style="list-style-type: none"> • Test cases for formal, executable parts of a MODEL SUBSET (e.g. state machines), which are intended to validate the behaviour specified in the MODEL SUBSET. These test cases e.g. can be implemented by textual lists with input stimuli and expected responses or by using SysML sequence diagrams. • Test cases for informal or semi-formal parts of a MODEL SUBSET (e.g. UseCases, block diagrams, flow specifications, etc.), which are intended for mostly manual quality assurance of a MODEL SUBSET. These test cases are usually less formal and consist e.g. of informal checklists.
Eu.SEP.62	Head	3.10 System test specification
Eu.SEP.63	Info	A SYSTEM TEST SPECIFICATION is a collection of test cases for a defined piece of kit (i.e. a physical system or a software component) which has been designed and built according to a FUNCTIONAL SPECIFICATION and/or one or more INTERFACE SPECIFICATIONS. The kit to be tested is usually called system under test (SuT).
Eu.SEP.67	Info	As of now, there are no strict requirements defined for the structure and contents of SYSTEM TEST SPECIFICATIONS within this document. As a general requirement, the SYSTEM TEST SPECIFICATION must be designed in a way that a sufficient acceptance test for a given kit can be performed by using it.
Eu.SEP.66	Info	<p>A SYSTEM TEST SPECIFICATION may be based on the formal part of the MODEL TEST SPECIFICATION for the same artefact. As a SYSTEM TEST SPECIFICATION is less abstract than a MODEL TEST SPECIFICATION, the latter usually needs to be extended by (non-complete list):</p> <ul style="list-style-type: none"> • Additional test cases due to the technical details of the kit's implementation • Telegram definitions according to INTERFACE SPECIFICATIONS • Configuration data according to data prep cluster for the SuT and other kits involved in the setup • Configuration data for the different layers of the protocol stack • Simulation of systems in the environment of the SuT, which are not physically present • Additional information and configuration data for automatic or manual test execution • Additional information for test documentation and management
Eu.SEP.64	Head	3.11 Validation checklist

ID	Type	Requirements
Eu.SEP.65	Info	<p>A VALIDATION CHECKLIST is a document created in the QM activities comprising comprehensive instructions on how to perform a validation for:</p> <ul style="list-style-type: none"> • FUNCTIONAL SPECIFICATIONS • INTERFACE DEFINITIONS • INTERFACE SPECIFICATIONS
Eu.SEP.68	Info	On this abstraction level, there is no strict definition on contents and format of a VALIDATION CHECKLIST. This needs to be agreed upon between the cluster leader within the QM activities and documented in a central document, e.g. [Eu.Doc.31]. The SPECIFICATION VALIDATOR uses this checklist to validate the specification documents.
Eu.SEP.69	Head	4 Stakeholders/Roles
Eu.SEP.70	Head	4.1 Cluster leader
Eu.SEP.71	Info	Cluster manager(s) responsible for the work results of a given EULYNX cluster.
Eu.SEP.72	Head	4.2 Cluster engineer
Eu.SEP.73	Info	Person(s) responsible for the actual engineering work on a cluster's MODEL SUBSET, FUNCTIONAL SPECIFICATION, INTERFACE DEFINITION or INTERFACE SPECIFICATION. The responsibility does not only cover the MODEL SUBSET itself, but also all documents based upon it.
Eu.SEP.223	Head	4.3 Model verifier
Eu.SEP.224	Info	The MODEL VERIFIER is responsible for verifying a MODEL SUBSET during the actual engineering work from the CLUSTER ENGINEER against a MODEL TEST SPECIFICATION (for verification).
Eu.SEP.74	Head	4.4 Model validator
Eu.SEP.75	Info	The MODEL VALIDATOR is responsible for validating a MODEL SUBSET against a MODEL TEST SPECIFICATION. It is highly recommended, that this role is not fulfilled by the same person who also acts as CLUSTER ENGINEER in the same cluster project. It is also highly recommended that the MODEL VALIDATOR is not only an expert for formal model validation, but also explicitly needs railway signalling expertise.
Eu.SEP.76	Head	4.5 System tester
Eu.SEP.77	Info	The SYSTEM TESTER is performing the tests of a physical kit or software artefact against the SYSTEM TEST SPECIFICATION.
Eu.SEP.78	Head	4.6 Specification validator
Eu.SEP.79	Info	The SPECIFICATION VALIDATOR uses the VALIDATION CHECKLIST to validate a FUNCTIONAL SPECIFICATION, an INTERFACE DEFINITION or an INTERFACE SPECIFICATION. It is highly recommended, that this role is not fulfilled by the same person who also acts as CLUSTER ENGINEER in the same cluster project.

ID	Type	Requirements
Eu.SEP.80	Head	5 Activities
Eu.SEP.81	Info	The following sub-sections describe the different activities in the MBSE process, as shown in the rows of Fig. 1 .
Eu.SEP.82	Head	5.1 Edit conceptual documents
Eu.SEP.83	Info	This activity covers editing the CONCEPTUAL DOCUMENTS as defined in section 3.8 and the current documentation list.
Eu.SEP.84	Info	If this activity is executed for the first time, the necessary CONCEPTUAL DOCUMENTS must be created and filled with initial content. If this activity is executed again, the existing CONCEPTUAL DOCUMENTS must be reviewed for necessary changes and changed accordingly. Changes might become necessary due to incremental additions, new findings or decisions regarding the CONCEPTUAL DOCUMENTS themselves. Changes might also become necessary to consistently adopt the CONCEPTUAL DOCUMENTS to changes of other artefacts in previous iterations.
Eu.SEP.85	Info	As CONCEPTUAL DOCUMENTS are textual documents with a large variety of purposes and content, no specific processes are defined for their creation or editing.
Eu.SEP.88	Info	Input artefacts: none Output artefacts: CONCEPTUAL DOCUMENTS Involved roles: CLUSTER ENGINEER
Eu.SEP.86	Head	5.2 Edit functional list
Eu.SEP.87	Info	This activity covers editing the FUNCTIONAL LIST as defined in section 3.1. The FUNCTIONAL LIST is the top-most functional artefact for every EULYNX cluster.
Eu.SEP.89	Info	If this activity is executed for the first time, the FUNCTIONAL LISTS must be created and filled with the initial content. If this activity is executed again, the existing FUNCTIONAL LISTS must be reviewed for necessary changes and changed accordingly. Changes might become necessary due to incremental additions, new findings or decisions regarding the FUNCTIONAL LISTS themselves. Changes might also become necessary to consistently adopt the FUNCTIONAL LISTS to changes of other artefacts in previous iterations.
Eu.SEP.92	Info	As functional lists are the top-most functional artefacts, changes are expected especially due to changes to the related MODEL SUBSET in previous iterations (see sections 5.3.1 and 5.3.2). The consistency of functional descriptions in the functional list as well as the applicability to IMs to the related MODEL SUBSET must be assured in this activity.
Eu.SEP.91	Info	Input artefacts: CONCEPTUAL DOCUMENTS Output artefacts: FUNCTIONAL LIST Involved roles: CLUSTER LEADER, CLUSTER ENGINEER
Eu.SEP.90	Head	5.3 MBSE activities

ID	Type	Requirements
Eu.SEP.93	Info	The following sub chapters describe the MBSE activities as core elements of the systems engineering process. As most of the MBSE activities as well as the process steps for modelling are described in detail in [Eu.Doc.30], this document will only reference to its relevant chapters.
Eu.SEP.94	Info	<p>MBSE activities are divided into three sub activities:</p> <ul style="list-style-type: none"> • Semi-formal subsystem definition using use cases and sequence diagrams (see section 5.3.1) • Formal logical architecture using state machines (see section 5.3.2) • Model validation using MODEL TEST CASES (see section 5.3.3)
Eu.SEP.95	Head	5.3.1 MBSE - subsystem definition
Eu.SEP.96	Info	This activity covers editing the MODEL SUBSET for a given subsystem or interface to generate a semi-formal subsystem or interface definition.
Eu.SEP.97	Info	If this activity is executed for the first time, the MODEL SUBSET will be created by using the informal system definition from architecture cluster and the corresponding FUNCTIONAL LIST. If this activity is executed again, the existing MODEL SUBSET must be reviewed for necessary changes and changed accordingly. Changes might become necessary due to incremental additions, new findings or decisions regarding the MODEL SUBSET itself. Changes might also become necessary to consistently adopt the MODEL SUBSET to changes of other artefacts in previous iterations.
Eu.SEP.100	Info	The MODEL may contain required by a number of MODEL SUBSETS. This applies e.g. for common functionality like booting and setup of communication. It is not recommended to model these generic elements multiple times in different MODEL SUBSETS as consistency cannot be assured.
Eu.SEP.98	Info	For generic parts of the model, it is highly recommended to create a SysML profile comprising them. This profile can be managed and updated independently from MODEL SUBSETS and reused by them. Technical details on how to use SysML profiles for generic parts of the model can be found in [MAINT, Chapter 6].
Eu.SEP.99	Info	<p>This activity is performed by creating/editing the following model-views in the MODEL SUBSET according to [Eu.Doc.30]:</p> <ul style="list-style-type: none"> • Technical system context - [Eu.Doc.30, chapter 9.2.2.1/9.2.2.2] • Functional system context - [Eu.Doc.30, chapters 9.2.2.3/9.2.2.4] • Information flow at the system interfaces - [Eu.Doc.30, chapters 9.2.2.5/9.2.2.6] • Essential states of the system - [Eu.Doc.30, chapters 9.2.2.8] • System UseCases - [Eu.Doc.30, chapters 9.2.2.9/9.2.2.10]
Eu.SEP.101	Info	<p>Input artefacts: FUNCTIONAL LIST</p> <p>Output artefacts: MODEL SUBSET</p> <p>Involved roles: CLUSTER ENGINEER</p>

ID	Type	Requirements
Eu.SEP.102	Head	5.3.2 MBSE - logical architecture
Eu.SEP.103	Info	This activity covers editing the MODEL SUBSET for a given subsystem or interface to generate formal subsystem or interface requirements implementing informal UseCases using logical components. Usually, this is done by specifying the behavioural aspects of logical components as state machines according to [Eu.Doc.30] (see below).
Eu.SEP.104	Info	If this activity is executed for the first time, the MODEL SUBSET will be created by using the semi-formal subsystem definition (see section 5.3.1) in the MODEL SUBSET. If this activity is executed again, the existing MODEL SUBSET must be reviewed for necessary changes and changed accordingly. Changes might become necessary due to incremental additions, new findings or decisions regarding the MODEL SUBSET itself. Changes might also become necessary to consistently adopt the MODEL SUBSET to changes of other artefacts in previous iterations.
Eu.SEP.105	Info	<p>This activity is performed by creating/editing the model-views in the MODEL SUBSET according to [Eu.Doc.30]:</p> <ul style="list-style-type: none"> • Logical Components [Eu.Doc.30, chapters 9.2.5.1/9.2.5.2] • Logical Architecture (System) [Eu.Doc.30, chapters 9.2.5.3/9.2.5.4] • Logical Architecture (System interfaces) [Eu.Doc.30, chapters 9.2.5.5/9.2.5.6] <p>Basically, the subsystem or interface requirements are specified by creating one or more logical components within the MODEL SUBSYSTEM and implementing the corresponding UseCases of the activity “MBSE – subsystem definition” (see section 5.3.1) as state machines.</p>
Eu.SEP.106	Info	<p>Input artefacts: MODEL SUBSET, FUNCTIONAL LIST</p> <p>Output artefacts: MODEL SUBSET</p> <p>Involved roles: CLUSTER ENGINEER</p>
Eu.SEP.225	Head	5.3.3 MBSE - model verification
Eu.SEP.226	Info	This activity covers verifying the MODEL SUBSET for a given subsystem or interface by creating a MODEL TEST SPECIFICATION (for verification) and verifying the MODEL SUBSET against it.
Eu.SEP.227	Info	Beginning with the sub phase “LogArch – logical architecture and validation” (see section 6.2.2), the MODEL SUBSET contains executable state machines that encapsulate a number of requirements. To verify the correctness of the state machines during the development, they need to be tested against a MODEL TEST SPECIFICATION (for verification).
Eu.SEP.107	Head	5.3.4 MBSE - model validation
Eu.SEP.108	Info	This activity covers validating the MODEL SUBSET for a given subsystem or interface by creating a MODEL TEST SPECIFICATION and validating the MODEL SUBSET against it.

ID	Type	Requirements
Eu.SEP.109	Info	Beginning with the sub phase “LogArch – logical architecture and validation” (see section 6.2.2), the MODEL SUBSET contains executable state machines that encapsulate a number of requirements. To validate the correctness of the state machines, they need to be tested against a MODEL TEST SPECIFICATION. The latter needs to be created and maintained independently of the logical architecture’s state machines and <i>must not</i> be generated - automatically or manually - out of them. It is highly recommended, that the MODEL TEST SPECIFICATION is created by a person (MODEL VALIDATOR) independent of the person creating and maintaining the state machines in the MODEL SUBSET (CLUSTER ENGINEER).
Eu.SEP.110	Info	The MODEL TEST SPECIFICATION shall be specified in a way, that all applicable requirements as well as basic principles of safe, efficient and reliable railway operation can be validated (see section 5.8.2.2 for basic information regarding validation).
Eu.SEP.111	Info	The MODEL TEST SPECIFICATION also comprises informal check lists for non-executable parts of the MODEL SUBSET (see section 3.9). These parts of the MODEL TEST SPECIFICATION must be created within this activity as well, and the semi-formal parts of the MODEL SUBSET checked against it accordingly.
Eu.SEP.112	Info	If this activity is executed for the first time, the MODEL TEST SPECIFICATION will be created. If this activity is executed again, the existing MODEL TEST SPECIFICATION must be reviewed for necessary changes and changed accordingly. Changes might become necessary due to incremental additions, new findings or decisions regarding the MODEL TEST SPECIFICATION itself. Changes might also become necessary to consistently adopt the MODEL TEST SPECIFICATION to changes of other artefacts in previous iterations.
Eu.SEP.113	Info	After creating or editing the MODEL TEST SPECIFICATION, the MODEL SUBSET must be validated against it. Technical details on the model validation will be described in a separate document.
Eu.SEP.115	Info	The result of the model validation is either “passed” (the MODEL SUBSET fulfils the MODEL TEST SPECIFICATION) or “failed” (it does not). A failed validation might be the result of errors either in the MODEL SUBSET or in the MODEL TEST SPECIFICATION or in both. Therefore, after a failed model validation, CLUSTER ENGINEER and MODEL TESTER should be assess the situation together and decide upon necessary changes. These changes are then implemented in the following iteration.
Eu.SEP.116	Info	Input artefacts: MODEL SUBSET Output artefacts: MODEL TEST SPECIFICATION Involved roles: CLUSTER ENGINEER, MODEL VALIDATOR
Eu.SEP.117	Head	5.4 Edit specification documents
Eu.SEP.118	Info	This activity covers editing the specification documents for a given subsystem or interface. Depending on the artefact, a FUNCTIONAL SPECIFICATION and one or more INTERFACE DEFINITIONS and INTERFACE SPECIFICATIONS are needed. These artefacts are maintained as DOORS modules.
Eu.SEP.119	Info	If this activity is executed for the first time, the FUNCTIONAL SPECIFICATION and/or INTERFACE DEFINITION and/or INTERFACE SPECIFICATION will be created. If this activity is executed again, the existing FUNCTIONAL SPECIFICATION and/or INTERFACE DEFINITION and/or INTERFACE SPECIFICATION must be reviewed for necessary changes and changed accordingly. Changes might become necessary due to incremental additions, new findings or decisions regarding the FUNCTIONAL SPECIFICATION and/or INTERFACE DEFINITION and/or INTERFACE SPECIFICATION themselves. These might also become necessary to consistently adopt the FUNCTIONAL SPECIFICATION and/or INTERFACE DEFINITION and/or INTERFACE SPECIFICATION to changes of other artefacts in previous iterations.

ID	Type	Requirements
Eu.SEP.120	Info	<p>The different type of specification artefacts are created by using different processes:</p> <ul style="list-style-type: none"> A FUNCTIONAL SPECIFICATION is created or edited by synchronising a MODEL SUBSET into the associated DOORS module and performing additional manual process steps according to [SYNC] afterwards. Only the synchronisation and manual post processing of the FUNCTIONAL SPECIFICATION are part of this process step; the functional changes to the associated MODEL SUBSET themselves are described in sections 5.3.1 and 5.3.2. An INTERFACE DEFINITION or INTERFACE SPECIFICATION is manually edited in DOORS.
Eu.SEP.121	Info	<p>Input artefacts: MODEL SUBSET</p> <p>Output artefacts: FUNCTIONAL SPECIFICATION, INTERFACE DEFINITION, INTERFACE SPECIFICATION</p> <p>Involved roles: CLUSTER ENGINEER</p>
Eu.SEP.122	Head	5.5 Validate specification documents
Eu.SEP.123	Info	This activity covers validating the specification documents for a given subsystem or interface. Depending on the artefact, a FUNCTIONAL SPECIFICATION and one or more INTERFACE DEFINITIONS and INTERFACE SPECIFICATIONS are validated. If possible, the FUNCTIONAL SPECIFICATION and the interface documents should be validated together to be able to validate consistency.
Eu.SEP.124	Info	The necessary validation steps are specified in the VALIDATION CHECKLIST. For parts of the specification documents that are generated out of the MODEL, the functional validation is covered by activity “MBSE – model validation” (see section 5.3.3). For any other part, the VALIDATION CHECKLIST must be used to manually check the validity of the specification documents.
Eu.SEP.125	Info	The VALIDATION CHECKLIST shall be specified in a way, that all applicable requirements as well as basic principles of safe, efficient and reliable railway operation can be validated (see section 5.8.2.2 for basic information regarding validation).
Eu.SEP.126	Info	The results of the validation must be documented in a protocol. Changes due to the validation are implemented into the artefacts in the following iterations.
Eu.SEP.127	Info	<p>Input artefacts: FUNCTIONAL SPECIFICATION, INTERFACE DEFINITION, INTERFACE SPECIFICATION, VALIDATION CHECKLIST</p> <p>Output artefacts: SYSTEM TEST SPECIFICATION</p> <p>Involved roles: SPECIFICATION VALIDATOR</p>
Eu.SEP.128	Head	5.6 Edit system test specification
Eu.SEP.129	Info	This activity covers editing the SYSTEM TEST SPECIFICATION for a given subsystem or interface. There is no requirement for a specific structure or format for the SYSTEM TEST SPECIFICATION yet. A SYSTEM TEST SPECIFICATION can be derived from the MODEL TEST SPECIFICATION according to section 3.10.

ID	Type	Requirements
Eu.SEP.130	Info	If this activity is executed for the first time, the SYSTEM TEST SPECIFICATION will be created by using the MODEL TEST SPECIFICATION (see section 5.3.2) for the associated MODEL SUBSET and extending it as necessary. If this activity is executed again, the existing SYSTEM TEST SPECIFICATION must be reviewed for necessary changes and changed accordingly. Changes might become necessary due to incremental additions, new findings or decisions regarding the SYSTEM TEST SPECIFICATION itself. Changes might also become necessary to consistently adopt the SYSTEM TEST SPECIFICATION to changes of other artefacts in previous iterations or due to failed system acceptance tests (see section 5.6).
Eu.SEP.131	Info	When completed, the SYSTEM TEST SPECIFICATION must fulfil the test targets agreed upon with system assurance cluster and documented in [Eu.Doc.31].
Eu.SEP.132	Info	Input artefacts: MODEL SUBSET Output artefacts: SYSTEM TEST SPECIFICATION Involved roles: CLUSTER ENGINEER, SYSTEM TESTER
Eu.SEP.133	Head	5.7 Perform acceptance tests
Eu.SEP.134	Info	This activity covers performing the system acceptance test for physical kits or software artefacts. As this is not a core SE activity, only the interface to the SE process is described. The acceptance test process itself is documented in [Eu.Doc.31].
Eu.SEP.135	Info	The acceptance test is based on the SYSTEM TEST SPECIFICATION for a given artefact. The result of the acceptance test is either “passed” (the kit passes the SYSTEM TEST SPECIFICATION) or “failed” (it does not). A failed test might be the result of either a wrong implementation of the kit and/or an error in the SYSTEM TEST SPECIFICATION. After a test fails, the SYSTEM TESTER needs to assess the root causes for the failed test. If the reason for the failed tests is an error in the SYSTEM TEST SPECIFICATION, this error must be fixed in the next iteration (see section 6.3 for details).
Eu.SEP.136	Info	Input artefacts: SYSTEM TEST SPECIFICATION Output artefacts: - Involved roles: SYSTEM TESTER
Eu.SEP.137	Head	5.8 Management activities
Eu.SEP.138	Head	5.8.1 Configuration management (CM)
Eu.SEP.139	Head	5.8.1.1 Variability management
Eu.SEP.140	Info	The EULYNX specifications contain variability. The only currently existing variation points are the applicability of requirements for particular IMs. As default, all requirements are applicable for all IMs. If a given requirement is only applicable for one or more IMs, the applicability for this requirement is restricted.
Eu.SEP.141	Info	Within the MODEL, restricted applicability is specified as described in [Eu.Doc.28].

ID	Type	Requirements
Eu.SEP.143	Info	In FUNCTIONAL SPECIFICATIONS, INTERFACE DEFINITIONS and INTERFACE SPECIFICATIONS the restricted applicability is indicated by DOORS attributes. The DOORS attributes are automatically generated for IDs within the module that are synced from a MODEL SUBSET (see section 3.4) by using the tool “VariSync”. For all other IDs, the applicability attributes must be set manually.
Eu.SEP.144	Info	See [Eu.Doc.28] for details on the technical aspects of variability management between MODEL and the DOORS-based specification modules.
Eu.SEP.145	Head	5.8.1.2 Model maintenance
Eu.SEP.146	Info	Technical information on maintaining the MODEL can be found in [MAINT].
Eu.SEP.147	Head	5.8.1.3 Baseline management
Eu.SEP.148	Info	Baseline management is defined to differentiate baseline versions of FUNCTIONAL SPECIFICATIONS, INTERFACE DEFINITIONS and INTERFACE SPECIFICATIONS in DOORS according to types of changes, identified in these three categories: Functional, Editorial, Applicability. The management of baseline versions is documented in [Eu.Doc.28].
Eu.SEP.149	Head	5.8.2 Quality management (QM)
Eu.SEP.150	Info	Quality management is a continuous activity running through all phases of the project. It comprises general quality management tasks defining QM-principles as well as specific tasks valid for selected artefacts, activities and phases.
Eu.SEP.151	Head	5.8.2.1 General QM tasks
Eu.SEP.152	Info	<p>The following best practices shall be implemented by the CLUSTER LEADER for a given cluster to assure sufficient quality of the specification artefacts:</p> <ul style="list-style-type: none"> • Always check for and use the latest version of templates for your final documents • All final documents must be reviewed at least by one person not involved in the creation of the document; reviews by multiple persons (e.g. one person per participating IM) are highly recommended • Try to automate routine tasks by use of macros, DXL scripts or similar means to avoid errors • At CLUSTER LEADERS discretion, use additional checklists for activities as needed
Eu.SEP.153	Head	5.8.2.2 Specific QM tasks
Eu.SEP.154	Info	The specific QM tasks consist of verification and validation efforts. While the verification efforts are implicitly defined per phase, the validation is realised by special validation activities. See below for details. All verification and validation activities must be coordinated with the Assurance cluster. If this leads to changes relevant for this document, they will be included in later versions.
Eu.SEP.157	Info	<p><u>Verification activities</u></p> <p>According to the CENELEC standards, the work results of each CENELEC phase must be verified against previous phase. As CENELEC phases are mapped to specific phases of this SE process, the description of the phases in section 6 contains also information about verification requirements. The CLUSTER LEADER shall ensure that these verification requirements are met.</p>

ID	Type	Requirements
Eu.SEP.158	Info	<p><u>Validation activities</u></p> <p>Validation in context of this SE process is defined as an activity to check whether an artefact is suitable to fulfil its purpose, complementary to already performed verification activities. For example, the formal parts of a MODEL SUBSET should be validated against the basic principles of safe and reliable railway operation and not only verified to fulfil the formal requirements (for details see section 6.2.2) coming from a previous phase.</p>
Eu.SEP.159	Info	<p>The following validation activities are defined:</p> <ul style="list-style-type: none"> • MBSE – model validation (see section 5.3.3) – validation of the MODEL SUBSET for requirements contained in the MODEL • Validate specification documents (see section 5.5) – validation of the specification documents for requirements not contained in the MODEL
Eu.SEP.160	Head	5.8.3 Safety management (SM)
Eu.SEP.161	Info	<i>Note: intentionally left blank. To be filled in later version of this document.</i>
Eu.SEP.162	Head	5.8.4 Change control management (CCM)
Eu.SEP.163	Info	The change control management process in EULYNX is defined by a workflow, indicating all involved participants in the process, the required decision steps and corresponding change request states maintained in the change request system JIRA. The change control management activity is documented in [CCM].
Eu.SEP.164	Head	6 Phases
Eu.SEP.165	Info	Phases are periods of time within the EULYNX project during which the characteristic of work in a cluster is generally unchanged and sufficiently different from work in a previous or following phase. The transition of one phase to the next is defined by a set of conditions that need to be met before the transition can be performed.
Eu.SEP.166	Info	A phase can be divided into multiple sub phases, if needed. Sub phases differentiate tasks within the phase more finely. The transition from one sub phase to the next is not defined by a strict set of conditions but left to the CLUSTER LEADERS.
Eu.SEP.167	Info	Every phase and sub phase can comprise one or more iterations. During every iteration, the activities (see section 5) defined for that phase (see description of phases below) must be performed in the given order.
Eu.SEP.168	Head	6.1 Concept phase
Eu.SEP.169	Info	This phase represents the starting point of work in a given cluster and aims at the documentation of the cluster's basic organisational and functional concepts. This comprises the creation and update of conceptual documents and functional lists, where the latter represent the basic functional definition for a subsystem or interface.
Eu.SEP.170	Info	This phase usually consists of multiple iterations, especially for the update of the functional lists. In every iteration, additional functions are added to the list and the conceptual documents are adapted accordingly.

ID	Type	Requirements
Eu.SEP.171	Info	The transition to the next phase may be performed, when the cluster members agree upon a version of the functional list that is considered complete and correct at that point of time.
Eu.SEP.172	Info	<p><u>Activities in phase:</u></p> <ul style="list-style-type: none"> Edit conceptual documents (section 5.1) – primary activity Edit functional lists (section 5.2) – primary activity <p><u>Transition condition to next phase:</u></p> <ul style="list-style-type: none"> Functional list comprises functionality agreed upon as “preliminary correct and complete” by cluster members. <p><u>Verification requirements:</u> none</p>
Eu.SEP.173	Head	6.2 Development phase
Eu.SEP.174	Info	<p>The development phase represents the core of the EULYNX development activities. To have finer control over the activities during that long lasting phase, it is further divided into three sub phases:</p> <ul style="list-style-type: none"> Sub phase “SubSysDef” (section 6.2.1) is focused on the semi-formal subsystem definition Sub phase “LogArch” (section 6.2.2) is focused on the formal system requirements Sub phase “TestGen” (section 6.2.3) is focused on generating test cases for kit acceptance tests <p>These sub phases are described more detailed in the following sub sections.</p>
Eu.SEP.175	Info	<p><u>Activities:</u> see sub phase description</p> <p><u>Transition condition to next phase:</u> The transition to the next development phase may be performed, when the following conditions are met:</p> <ul style="list-style-type: none"> MODEL SUBSET completed and successfully validated Specification documents (FUNCTIONAL SPECIFICATION, INTERFACE DEFINITION and/or INTERFACE SPECIFICATION) successfully generated and edited SYSTEM TEST SPECIFICATION successfully created and validated Verification requirements are met <p><u>Verification requirements:</u></p> <ul style="list-style-type: none"> MODEL SUBSET and the specification documents (FUNCTIONAL SPECIFICATION, INTERFACE DEFINITION and/or INTERFACE SPECIFICATION) fulfil requirements of FUNCTIONAL LIST, SYSTEM DEFINITION and applicable CONCPETUAL DOCUMENTS SYSTEM TEST SPECIFICATION is adequate to perform acceptance tests for the functions defined in FUNCTIONAL LIST
Eu.SEP.176	Head	6.2.1 SubSysDef - Subsystem definition

ID	Type	Requirements
Eu.SEP.178	Info	During this sub phase, the semi-formal subsystem/interface definition for a given subsystem and/or interface is created in multiple iterations. Iterations should be structured in a way that a subset of the actors, their interfaces and information flows from the SYSTEM DEFINITION and the associated functions from the FUNCTIONAL LIST are implemented per iteration.
Eu.SEP.179	Info	This is done by performing activity “MBSE – subsystem definition” onto the existing SYSTEM DEFINITION and the FUNCTIONAL LIST from the previous phase. The system definition is created within the corresponding MODEL SUBSET. After the subsystem definition has been created or edited, the MODEL SUBSET is validated according to the activity “MBSE – model validation” in every iteration. As there are not yet formal parts of the model, the validation in this phase only covers semi-formal validation steps according to the activity definition. Both activities mentioned above are the primary activities during that sub phase.
Eu.SEP.180	Info	The activities “Edit conceptual documents” and “Edit functional lists” are also performed during every iteration to update the function lists and conceptual documents to new findings discovered during modelling work.
Eu.SEP.181	Info	Also part of this phase is the generation/creation of specification documents. This is covered by the activities “Edit specification documents” and “Validate specification documents”. These activities are performed in every iteration after the model has been edited to generate a set of documents representing the current state of the MODEL SUBSET.
Eu.SEP.182	Info	<p><u>Activities in phase:</u></p> <ul style="list-style-type: none"> Edit conceptual documents (see section 5.1) Edit functional lists (see section 5.2) MBSE – subsystem definition (see section 5.3.1) – primary activity MBSE – model validation (see section 5.3.3) – primary activity Edit specification documents (see section 5.5) Validate specification documents (see section 5.6)
Eu.SEP.183	Head	6.2.2 LogArch - logical architecture and validation
Eu.SEP.185	Info	During this sub phase, the formal parts of the MODEL SUBSET for a given subsystem and/or interface is created in multiple iterations. Iterations should be structured in a way that a subset of the use cases and their sequence diagrams are implemented as state machines per iteration.
Eu.SEP.184	Info	This is done by performing activity “MBSE – logical architecture” onto the existing semi-formal subsystem/interface definition in the MODEL SUBSET from the previous phase and by using remaining input from the FUNCTIONAL LIST. The system behaviour is modelled as state machines in the MODEL SUBSET. After the functionality has been modelled, the MODEL SUBSET is validated by performing activity “MBSE – model validation” onto both parts of the MODEL SUBSET (semi-formal and formal) according to the activity definition. Both activities mentioned above are the primary activities during that sub phase.
Eu.SEP.186	Info	As in the previous phase, the activities “Edit conceptual documents” and “Edit functional lists” are also performed during every iteration to update the function lists and conceptual documents to new findings discovered during modelling work. Respectively, the activity “MBSE – subsystem definition” is performed to update the semi-formal parts of the MODEL SUBSET to the current state of the formal parts of the MODEL SUBSET.

ID	Type	Requirements
Eu.SEP.187	Info	This phase again also comprises is the generation/creation of specification documents. This is covered by the activities “Edit specification documents” and “Validate specification documents”. These activities are performed in every iteration after the model has been edited to generate a set of documents representing the current state of the MODEL SUBSET. In this phase, this results in the specification documents being extended by the formal behaviour specification.
Eu.SEP.188	Info	<u>Activities in phase:</u> <ul style="list-style-type: none"> Edit conceptual documents (see section 5.1) Edit functional lists (see section 5.2) MBSE - subsystem definition (see section 5.3.1) MBSE - logical architecture (see section 5.3.2) - primary activity MBSE - model validation (see section 5.3.3) - primary activity Edit specification documents (see section 5.5) Validate specification documents (see section 5.6)
Eu.SEP.189	Head	6.2.3 TestGen - generation of system test specification
Eu.SEP.191	Info	The focus of this phase is not on developing the MODEL SUBSET any further, but on the creation of the associated SYSTEM TEST SPECIFICATION. The basic concept is to (at least partially) generate this SYSTEM TEST SPECIFICATION out of the formal parts of the MODEL SUBSET automatically. Therefore, the activity mentioned above is the primary activity during that sub phase.
Eu.SEP.192	Info	This is done in a widely established process, where a state machine is analysed with a given coverage criterion (e.g. full state coverage or full transition coverage) and a number of test sequences in generated as a result. Details are described in [Eu.Doc.31]. As the state machines usually do not contain enough technical details, the generated test sequences must be amended manually to reach sufficient level of detail for a SYSTEM TEST SPECIFICATION (see also section 3.10).
Eu.SEP.193	Info	The main task in this phase is performing the activity “Edit system test specification” iteratively. Iterations should be structured in a way, that per iteration test cases for a defined subset of the state chart functionality are generated. This can be done e.g. by orientation on the sub system’s use cases as a structuring property.
Eu.SEP.194	Info	The other activities in this phase are performed to eventually adopt all other artefacts to changes introduced while generating the SYSTEM TEST SPECIFICATION. Usually, the generation of test cases itself will not alter other artefacts. However, it cannot be ruled out, that during test case generation errors and omissions will be found in other artefacts that need to be corrected.
Eu.SEP.195	Info	<u>Activities in phase:</u> <ul style="list-style-type: none"> Edit conceptual documents (see section 5.1) Edit functional lists (see section 5.2)

ID	Type	Requirements
Eu.SEP.195		<ul style="list-style-type: none"> • MBSE – subsystem definition (see section 5.3.1) • MBSE – logical architecture (see section 5.3.2) • MBSE – model validation (see section 5.3.3) • Edit specification documents (see section 5.5) • Validate specification documents (see section 5.6) • Edit system test specification (see section 5.5) – primary activity
Eu.SEP.196	Head	6.3 System acceptance phase
Eu.SEP.197	Info	During this phase, the system acceptance tests for a given kit or software artefact are performed. This phase will be repeated every time an acceptance test for another kit is performed. Note: this phase might at least partially run in parallel to the phase 6.4 - Maintenance phase.
Eu.SEP.198	Info	In this phase the kit or software artefact is tested against the SYSTEM TEST SPECIFICATION according to activity “Perform acceptance tests” (see section 5.7). To keep the testing process manageable, this phase can comprise multiple testing iterations, each testing a defined feature set of the kit or software artefact.
Eu.SEP.199	Info	After each testing iteration, the failed tests need to be analysed according to the aforementioned activity. If the acceptance test revealed errors in the specification, these must be corrected by using the change management process as defined in [Eu.Doc.28]. Therefore, it might become necessary perform any of the defined activities to correct the errors in the specification and to make it consistent again.
Eu.SEP.200	Info	<p><u>The following activities are mandatory in every iteration of this phase:</u></p> <ul style="list-style-type: none"> • Perform acceptance tests (see section 5.7) – primary activity
Eu.SEP.201	Info	<p><u>The following activities are only to be performed, if a change request has been raised and has been approved by the CCM:</u></p> <ul style="list-style-type: none"> • Edit conceptual documents (see section 5.1) • Edit functional lists (see section 5.2) • MBSE – subsystem definition (see section 5.3.1) • MBSE – logical architecture (see section 5.3.2) • MBSE – model validation (see section 5.3.3) • Edit specification documents (see section 5.5) • Validate specification documents (see section 5.6) • Edit system test specification (see section 5.5)

ID	Type	Requirements
Eu.SEP.202	Info	<u>Transition condition to next phase:</u> <ul style="list-style-type: none"> • All approved CRs have been implemented • MODEL SUBSET and specification documents have been validated • SYSTEM TEST SPECIFICATION has been updated and is consistent to specification documents • All tests of the kit or software artefact are passed
Eu.SEP.203	Info	<u>Verification requirements:</u> <ul style="list-style-type: none"> • MODEL SUBSET and the specification documents (FUNCTIONAL SPECIFICATION, INTERFACE DEFINITION and/or INTERFACE SPECIFICATION) fulfil requirements of FUNCTIONAL LIST, SYSTEM DEFINITION and applicable CONCPETUAL DOCUMENTS • SYSTEM TEST SPECIFICATION is adequate to perform acceptance tests for the functions defined in FUNCTIONAL LIST
Eu.SEP.204	Head	6.4 Maintenance phase
Eu.SEP.205	Info	This phase runs as long as EULYNX kits or software components are operational at IMs. It deals with maintaining the specification over time and to include change requests. Change requests might arise due to erroneous behaviour of a kit in the field as well as due to new feature requests.
Eu.SEP.206	Info	Change requests are managed according to [Eu.Doc.28]. If a CR has been approved, it might become necessary perform any of the defined activities to implement the change request into the specification and to make it consistent again.
Eu.SEP.207	Info	<u>The following activities are to be performed, if a change request has been raised and has been approved by the CCM:</u> <ul style="list-style-type: none"> • Edit conceptual documents (see section 5.1) • Edit functional lists (see section 5.2) • MBSE – subsystem definition (see section 5.3.1) • MBSE – logical architecture (see section 5.3.2) • MBSE – model validation (see section 5.3.3) • Edit specification documents (see section 5.5) • Validate specification documents (see section 5.6) • Edit system test specification (see section 5.5) • Perform acceptance tests (see section 5.7)
Eu.SEP.208	Info	<u>Transition condition to next phase:</u> none

ID	Type	Requirements
Eu.SEP.209	Info	<u>Verification requirements:</u> <ul style="list-style-type: none"> MODEL SUBSET and the specification documents (FUNCTIONAL SPECIFICATION, INTERFACE DEFINITION and/or INTERFACE SPECIFICATION) fulfil requirements of FUNCTIONAL LIST, SYSTEM DEFINITION and applicable CONCPETUAL DOCUMENTS SYSTEM TEST SPECIFICATION is adequate to perform acceptance tests for the functions defined in FUNCTIONAL LIST
Eu.SEP.210	Head	⁷ Process instances
Eu.SEP.211	Info	This process is designed to be instantiated once per cluster project that creates at least one of the artefacts MODEL SUBSET, FUNCTIONAL SPECIFICATION, INTERFACE DEFINITION or INTERFACE SPECIFICATION.
Eu.SEP.212	Info	This reflects the necessity to make the development of the different clusters independent from one another.