 THE RESEARCH LEADING TO THESE RESULTS HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S SEVENTH FRAMEWORK PROGRAM (FP7/2007-2013) FOR CRYSTAL – CRITICAL SYSTEM ENGINEERING ACCELERATION JOINT UNDERTAKING UNDER GRANT AGREEMENT N° 332830 AND FROM SPECIFIC NATIONAL PROGRAMS AND / OR FUNDING AUTHORITIES.

CRitical S**YST**em Engineering **A**cce**L**eration

Specification, Development and Assessment for Product Lifecycle Management

D608.011

DOCUMENT INFORMATION

Project	CRYSTAL
Grant Agreement No.	ARTEMIS-2012-1-332830
Deliverable Title	Specification, Development and Assessment for Product Lifecycle Management
Deliverable No.	D608.011
Dissemination Level	CO
Nature	R
Document Version	3.0
Date	2014-02-05
Contact	Matthias Frische
Organization	SISW
Phone	+49 (160) 7180163
E-Mail	matthias.frische@siemens.com

AUTHORS TABLE

Name	Company	E-Mail
Matthias Frische	SISW	matthias.frische@siemens.com
Michael Hollenstein	SISW	michael.hollenstein@siemens.com
Lan Liu	SISW	lan.liu@siemens.com
Rubén de Juan Marín	ITI	rjuan@iti.es

CHANGE HISTORY

Version	Date	Reason for Change	Pages Affected
1.00	2013-10-25	Initial version	all
1.10	2014-1-27	Major rewrite	all
2.0	2014-1-28	SISW Internal Review – for upload	all
3.0	2014-2-4	Internal and External Review - corrections	many

CONTENT

Content of Figures

1 Introduction

1.1 Role of deliverable

The role of this deliverable is to collect and analyse the Product Lifecycle Management requirements of an advanced systems engineering environment.

One of the tasks of WP 608 is to describe which PLM focused “engineering methods” need to be supported by an IOS/RTP interface. This will be the basis for the functional demands that guide the specification and technical implementation of this interface.

SP 6 needs this information as input to their work on the IOS/RTP specification process.

As one outcome of the specification process of the CRYSTAL IOS/RTP the selection and definition of technologies (like OSLC etc.) to implement the interface is done.

For clarification purposes the typical lifecycle capabilities are listed in this work package and are described in a short manner. In addition the requirements for the tools in the tool chains are described.

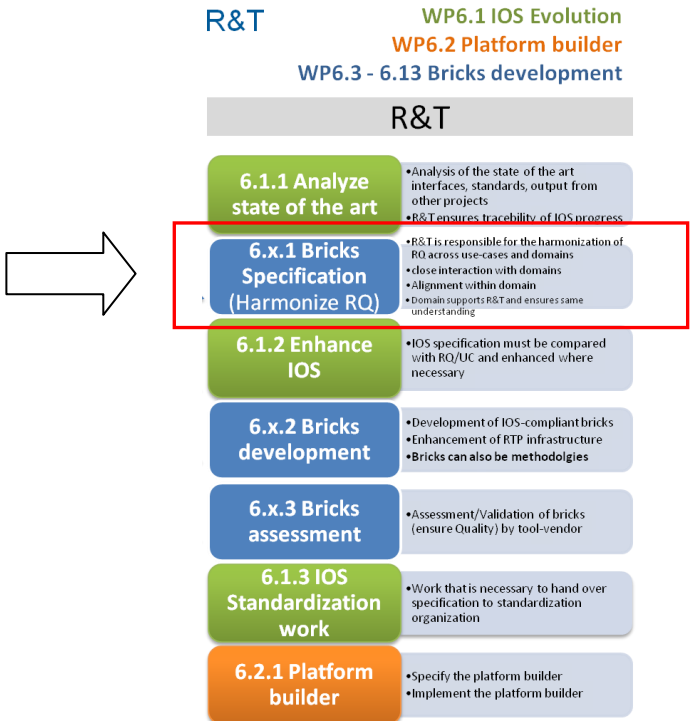


Figure 1-1: Role of the Document

1.2 Relationship to other CRYSTAL Documents

This document refers the full project proposal.

1.3 Structure of this document

This document is organized as follows:

- **Chapter 1**. Introduction
- **Chapter 2**.
- High-Level PLM Requirements
- Domain Specific PLM related Needs
- Engineering Methods
- PLM Tool Requirements
- **Chapter 3**. Terms, abbreviations and definitions
- **Chapter 4**. References

2 PLM Requirements Collection

2.1 High-Level PLM Requirements

Based on the interaction with the other CRYSTAL project members and the WP 608 participants a list of PLM related requirements for an IOS/RTP implementation was compiled.
This list also reflects the results of the CRYSTAL consolidation workshop from October 2013.

- Requirements Management
- Check Standard Compliance
- Lifecycle / Workflow Management (Maturity level)
- Validity, versions management, revisions of Artifacts
- Change Management and Traceability of changes
- Project management and Schedule Management
- Organizational structure: groups and roles
- Security and access management: access rules / group permissions etc. Roles and rights management,
- Data exchange: frequency and format of data exchange
- BOM-Management
- Consistency management
- Product Variant & Configuration Management
- Artifact Management
- Ontology Management
- Reporting & View Management
- Documentation Generation (Dashboards): the documentation of the progress of the systems engineering process
- Manage view from different directions (Requirements View / Functional View / Safety View etc.)

2.2 Domain Specific PLM related Needs

2.2.1 Approach to analyse the PLM specific Domain needs

Many of the PLM related requirements needed in the CRYSTAL IOS/RTP Interface/Platform can be extracted from the use cases that the domains supply, that are contributing to the CRYSTAL project. This of course does not prove that this collection of requirements is sufficient for the IOS/RTP definition and for a technical specification that can be successfully implemented. It is a fact that most of the use cases dive deep into the domain specific use of software tools and focus on the solution of sometimes very specific engineering problems and technology solutions. Therefore it is not always considered that the solution of an engineering problem might be only a part of the design process of the whole system or product.

This design process is embedded in industry specific commercial and regulatory environments.

For example a project management environment needs to be in place to manage risks (technical / commercial) and manage time constraints in terms of finding a solution in a limited amount of time.

This is often not part of the more technical description of the provided use cases.

The use case and domain specific IT infrastructures (i.e. access right management, global data distribution) can strongly impact the practical use of the open systems engineering environment that the participants of CRYSTAL want to achieve. While this might not be important for a specific use case this is important from a PLM perspective.

In the further sections the domains that are present in CRYSTAL are analysed and the processes and tools are looked at. The elaboration of further details takes place in later parts of this document. Due to the limited resources a subset of the whole requirement details can be analysed and can be used as input for the implementation of IOS/RTP. The prioritization process needs to take that into consideration.

As a result of this chapter a summary is created that reports the commonalities of the requirements of the different domains. This is the basis for a prioritization of the requirements from all domains. These common requirements will have to be implemented in the IOS/RTP Platform/Interface.

2.2.2 Aerospace

The analysis of this domain specific use case needs to be continued in order to confirm the findings currently used in the summary section of the Domain Specific PLM related needs. Depending on the need for further detailing and the available project resources this will be done in later stages of the project. The current description of the use case PLM related needs derives from the CRYSTAL project participant interaction and the requests for information that was answered by the WP 608 participants.

2.2.2.1 PLM related use cases

The Table below maps the engineering workflow steps of the User Stories to PLM related use cases The PLM related use cases are derived in the high-level use case list above. The high-level PLM use cases are functional driven and will be mapped to PLM related Bricks. So the table below supports the mapping

User Story -> Engineering Workflow -> PLM Related System Use Cases -> Solution Bricks

User story	User story title	User story objectives	Engineering workflow description	Step #	Requirements Management	Check Standard Compliance	Lifecycle / Workflow Management (Maturity level)	Validity, versions management, revisions of Artifacts	Change Management and Traceability of changes	Project management and Schedule Management	Organizational structure: groups and roles	Security and access management: access rules / group	Data exchange: frequency and format of data exchange	BOM-Management	Consistency management	Product Variant & Configuration Management	Artifact Management	Ontology Management	Reporting & View Management	Documentation Generation	Manage view
US2.01	performance and scheduling analysis	Integrate in the model scheduling and partitioning capabilities to perform performance and scheduling analysis in early design phases	Definition of scheduling by model based approach at system architecture level	1	x		x										x				
			Analyse Performance Constraints (Timings, Latency, Execution times, Sequences) on scheduling model	2	x																
			Refine candidate solution architecture or scheduling if necessary	3																	
			Possibly derive Operating System configuration after scheduling has been finalized	4																	
US2.02	Safety Analysis	Define a modelling approach to supportsafety analysis	Define functions of system under design	1	x												x				
			Define failure cases for the system functions	2	x												x				
			Define criticality of functions	3	x												x				

			Define candidate architecture to implement system functions with failure rates for candidate system components	4													x				
			Analyze quantitative safety aspects	5	x												x				
			Refine / change architecture if safety objectives have not been met	6	x	x											x				
US2.03	Variability Management	Define a process for the identification, specification and tracing of variability points to development artifacts	Model common and variable feature dependencies	1	x		x	x							x	x				
			Create a variability model covering external and internal variability	2			x	x							x	x				x
			Link the variability model with Requirements	3	x		x	x							x	x				x
			Link the variability model with System / Safety Architecture and Design Model, including SW	4	x		x	x							x	x				x
			cases and Procedures	5																	
			Develop variability model configurations to provide consistent visibility to external customer features and to external and internal variability.	6			x	x			x			x	x	x					
			create and manage components and systems model library and combine these models together.	7			x	x					x								
US2.04	Ontology based Requirements Formalization and validation	Define an approach for requirements formalization using boilerplates and domain ontologies	Develop domain ontology	1	x												x				x

			Define boilerplates	2			x									x				
			Capture requirements	3	x		x					x								
			Validate and update domain ontology	4				x									x			
			Define patterns (i.e. for formalizing contracts)	5			x									x				
			Analyse quality of requirements set	6	x			x												
			Manage the traceability between requirements and artifacts	7	x		x									x				
US2.05	Process automation, Guidance & Monitoring	Create a company related engineering workflow based on a process engineering meta-model	Define generic engineering artifacts (tasks, roles, rules, ...) in libraries	1						x	x	x				x				
			Build or tailor the specific company processes using the generic engineering libraries	2			x									x				
US2.06	Artifacts advanced Traceability	Set up a full traceability between the artifacts of the product development	Specify traceability meta model	1	x		x						x							
			Create requirements traceability links	2	x		x									x				x
			Analyse requirements using traceability links (validation, verification, change management)	3		x	x	x	x							x				x

			Support certification by traceability-based project monitoring	4		x	x	x									x					
US2.07	Component-based development	Set up a model based component approach	definition of system requirements	1	x		x										x					
			identification and specification of components	2			x										x					
			modelling of components relying on the contract based approach	3			x										x					
			Validate the component using simulation and proof of properties technologies	4	x		x	x									x					
			Compose the components to build up the appropriate functions and system	5	x		x										x					
			Validate the compositional model based on testing activities and formal proof techniques.	6			x	x														
US2.08	Multi-physics modelling and simulation	Support Holistic modelling for safety analysis	Derive from the requirements the safety and the functional views	1	x		x												x		x	
			Model the system environment	2			x															
			Identify the failure conditions	3	x																	
			Define the system inputs associated to failure conditions	4	x																	
			Simulate the system	5			x										x					

			Analyse the impact of the failure conditions on the system behaviour based on simulation results	6			x													
US2.09	A multi-views ontology in the aeronautics supply chain	Define aerospace ontology to exchange information in a model based development	Identify the data to be shared through collaborative workshops	1			x				x					x				x
			Model these data using multi-views ontology approach	2			x									x	x			
			Make the links between the data of the ontology and the models using these data	3			x									x	x			
			Populate the ontology and make an ontology baseline	4			x									x	x			
			Export data ontology to the different views	5								x					x			x
			Update the ontology from one view	6			x										x			x
			Make an impact analysis on the other views	7													x			x
			Update the other views	8													x			x
			Verify the models	9		x	x													
US2.10	Architecture exploration and trade-off	develop an operational methodology and tool chain for supporting architecture exploration and trade-offs.	Identify the different views, the exploration criteria and goals	1			x						x			x				x
			Identify a baseline	2			x						x			x				
			Model the system	3			x									x				
			Make the relevant analysis on the different views	4			x	x										x		x
			Analyse the analysis results	5			x													
			Choose the “best solution”	6			x									x				
			If the analysis results show the system design does not meet the criteria, modify the system design and make the analysis again until an appropriate solution is found.	7	x	x														
			Identify the design space	8			x									x				

US2.11	Automatic test case generation from formalized requirements	Generate test vectors from pattern-based formal requirements	Capture requirements using natural language or boilerplates	1	x		x									x													
			Capture, analyse requirements and convert them to pattern-based formal specifications	2	x		x									x													
			Automatically generate tests from pattern-based formal specifications	3	x		x									x													
			Execute the tests against the SUT (System Under Test)	4			x									x													
			Verify whether the requirements are violated	5	x	x	x									x													
			Measure requirements coverage and structural coverage	6	x		x											x		x		x							
Sum:					28	6	45	13	1	1	1	3	2	4	1	5	40	9	3	-	15								

2.2.2.2 Alenia related topics

Apart from the table above, Alenia asked for consideration of further topics the following topics with Alenia and will detail that further in the next deliverables:

- Alenia has requirements related to the interoperability between Teamcenter and Rhapsody
- Alenia describes the following PLM topics that need to be implemented in the CRYSTAL IOS RTP
  - System view (as required; as conceived) and system view description
  - Applicable process specification
  - Requirements based progress monitoring
  - Traceability among requirements and verification cases (model level and SUT level)
  - Single point access to RTP for users with identification of role.
  - Display and tracing of relevant events, such as updates to configuration and so on
  - Adoption of STEP compliant terminology for PLM
- Furthermore, we need to look at Alena´s needs for interoperability between PLM (as a Brick) and other ALM or creation tool applications to decide which technology for implementation should be chosen.

2.2.2.3 Processes

To be added

2.2.2.4 Tools

To be added

2.2.3 Automotive

The analysis of this domain specific use case needs to be continued in order to confirm the findings currently used in the summary section of the Domain Specific PLM related needs. Depending on the need for further detailing and the available project resources this will be done in later stages of the project. The current description of the use case PLM related needs derives from the CRYSTAL project participant interaction and the requests for information that was answered by the WP 608 participants.

2.2.3.1 PLM related use cases

To be added

2.2.3.2 Processes

To be added

2.2.3.3 Tools

To be added

2.2.4 Health Care

The analysis of this domain specific use case needs to be continued in order to confirm the findings currently used in the summary section of the Domain Specific PLM related needs. Depending on the need for further detailing and the available project resources this will be done in later stages of the project. The current description of the use case PLM related needs derives from the CRYSTAL project participant interaction and the requests for information that was answered by the WP 608 participants.

2.2.4.1 PLM related use cases

To be added

2.2.4.2 Processes

To be added

2.2.4.3 Tools

To be added

2.2.5 Rail

The analysis of this domain specific use case needs to be continued in order to confirm the findings currently used in the summary section of the Domain Specific PLM related needs. Depending on the need for further detailing and the available project resources this will be done in later stages of the project. The current description of the use case PLM related needs derives from the CRYSTAL project participant interaction and the requests for information that was answered by the WP 608 participants.

2.2.5.1 PLM related use cases

To be added

2.2.5.2 Processes

To be added

2.2.5.3 Tools

To be added

2.2.6 Summary / Common Requirements

About common requirements

Common requirements are the ones that are requested by one or more domains in the CRYSTAL project. Some of these requirements might not be requested by a domain but are visible as embedded requests for certain interaction and query features in their use cases. To analyse and extract these is part of WP 608. Some of the common requirements are requested by companies that are not present in the CRYSTAL project but that are customers of the commercial PLM vendors. These requirements have to be implemented into the CRYSTAL IOS/RTP solution in order to get acceptance by the commercial software vendors. In the case of the CRYSTAL IOS/RTP standard it is required that the CRYSTAL IOS/RTP solution helps to simplify the systems engineering work that needs to be done. The verification of common requirements will be achieved by the interaction of WP608 with the other work packages of the CRYSTAL project.

About prioritization

Prioritization is required in order to prevent that the implementation of the CRYSTAL IOS/RTP ends in an over-complexity of the specification. Risks exist that too many requested features lead to unmanageable implementation complexity.

In general an open standard as the CRYSTAL IOS/RTP enabled engineering environment needs to provide key components (features) that are acceptable and required by all participating parties. Even the parties that are not participating in the CRYSTAL project should be considered and reaching out to these is recommended by the work package 608 lead. Ignoring of valid input from external sources might cause a reduced acceptation of the interface/communication standard that the CRYSTAL project is working on to create.

About Openness

The CRYSTAL IOS/RTP interface implementation for a systems engineering environment has to be open enough to support specific components / features that are needed to solve the needs for very specialized domain specific requirements. Due to the wide variety of technological challenges especially in the environment of systems engineering an extensibility of the standard functionality is a non-negotiable must.

For commercial PLM software system vendors the Codex of PLM Openness

(http://www.prostep.org/fileadmin/user_upload/ProSTEPiViP/Profil/ProSTEP-iViP_CPO_V1_120308.pdf) provides a guideline for the openness of their software solutions and implementations of customer specific solutions. A glance at the rules that define openness according to the Codex of PLM Openness is helpful for the specification and realization of the CRYSTAL IOS/RTP project of establishing a interfacing definition and communication standard for advanced systems engineering environments.

Openness is not achieved by publishing a definition or defining a new standard that is communicated publicly. Openness is about keeping data not captured and providing easy methods to access the data or information.

2.3 Engineering Methods

Introduction and concerns

Siemens PLM Software and other participants of WP 608 see the challenge to clearly separate the functional description of engineering methods from the technicalities of the implementation (IT-technology). We are trying to apply a very formal description of “required engineering methods mandatory for PLM” in order to have the input for the SP 6 IOS/RTP specification as clear as possible. In WP 602 Platform Builder Alenia points out a similar topic, because in Platform Builder exists the same need to formalize "Engineering Methods" and "Engineering Activities" in order to have a mapping between "Engineering Activities" and "Engineering Tool Functions" (this is a task of Platform Builder workflow). In WP 602 Engineering Function represents a detailed function of an Activity and it refers to an action performed by means of a Tool. (for more details refer to:

https://projects.avl.com/11/0154/Data%20Exchange/007_Work/SP6_RnT_Activities/6.2%20Platform%20Builder/material/PlatformBuilder-Overview-IOS.pptx

The following questions exist: What is the granularity of Engineering Activities? How can we detail an Engineering Method through Engineering Activities? Interaction between WP 602 and IOS will address this topic: Identify a common template to describe Engineering Methods and defining also a common level of detail for Engineering Activities. Alenia pointed out that probably WP 608 should be interested about the "common level of detail" of Engineering Activities (as represented in the Engineering Methods Template).

In conclusion it is necessary that mainly WP 602, WP 608 and IOS, have to discuss this topic.

From the IOS Workshop “WP 2.08: Public Use Case Aerospace”, we got the input to define the Engineering Methods described in the following table. [IOS Workshop]

Engineering Method example: Analyze Requirement

Number	Process Activitiy	Engineering Methods	Tools Involved
1	Define Initial Requirements	• Analyze Requirement	RAT, DOORS
2	Analyze Requirements	• Analyze Requirement • Verify Design against Requirements	DQA/RQA, kM, Doors,
3	Preliminary concept definition and Trade-off analysis	*All below*	Papyrus, Rhapsody, FT+, Matlab Simulink, PTC Windchill Quality Solution, Altarica tool (TBD), Dymola,
3a	Define different alternative concepts for deicing system	• Maintain consistency between multi-viewpoint models	All above
3b	*Analyze and test conceptual models*	*Below*	All above
3b1	Perform static checks against different viewpoints	• Verify Design against requirements • Trade-off Analysis • Generate Fault-trees	All above (TBC)
3b2	Define and run simulation models	• Heterogeneous Simulation • Trade-off Analysis	Rhapsody, Matlab Simulink, Dymola, Altarica Tool
3c	Visualize all results in one "design exploration" tool	• Trade-off Analysis	TBD
4	Detailed concept definition and Trade-off analysis	*All below*	Papyrus, Rhapsody, FT+, Matlab Simulink, PTC Windchill Quality Solution, Altarica, Dymola, In-House Safety Tool
4a	Select a few concepts and define more detailed models for these concepts	• Maintain consistency between multi-viewpoint models	All above
4b	Analyze and test detailed models	• Verify Design against requirements • Trade-off Analysis • Heterogeneous Simulation • Generate Fault-trees	All above
4c	Visualize all results in one "design exploration" tool	• Trade-off Analysis	TBD
5	Select a concepts and define a specification for a supplier for one of the system components	• Provide specification document	TBD (Possible Candidates: Doors, Rational Publishing Engine)
6	Define and analyse a solution for the specified system component	• Provide specification document	Papyrus, Rhapsody, FT+, Matlab Simulink, PTC Windchill Quality Solution, Altarica, Dymola
7	Define a simulation model for system component, and run co-simulation	• Heterogeneous Simulation	Papyrus, Rhapsody, FT+, Matlab Simulink, PTC Windchill Quality Solution, Altarica, Dymola
T1	Provide Process Management	• Process Management	TBD
T2	Put all data under configuration control	TBD	Siemens TeamCenter, IBM RELM, RTC
T3	Show traceability between all data, e.g. for change impact analysis	• Change Impact analysis • Traceability Matrix	Siemens TeamCenter, IBM RELM, RTC
T4	Search Data	• Search Data	TBD

We want to extend this idea and define the most common Engineering Methods from PLM perspective. This list is the first draft of PLM Engineering methods:

- Define requirements,
- Analyse requirements
- Change requirements
- Verify Design against Requirements
- Create engineering and product object based on the requirements
- Maintain data consistency between requirements and the related objects
- Enable the traceability between the related objects and their textual information
- Enable to create and maintain the maturity of the objects during the whole lifecycle and maintain the maturity consistency
- Enable the schedule management during the whole process
- Enable change possibility and maintain the consistency

2.4 PLM Tool Requirements

When analyzing the PLM relevant engineering methods, not all of the IOS Concerns will be described in this document version but we start with a few basic and important aspects like Requirements Management and Lifecycle / Workflow Management. In the area of product variance and lifecycle maturity states like “In Work, Locked, Pre Released, Released, Closed” are present. These terms are not always present for the everyday detailed work of an engineer, but from a company perspective the tracking and tracing of revisions, workflows and completion levels is an important part of creating a structured environment that is needed for more efficient engineering processes. Some of the PLM tool requirements that are detailed here are also listed in the High-Level PLM requirements section.

2.4.1 Requirements Management and Check Standard Compliance

This section describes the needs when managing requirements.

2.4.1.1 Managing different sources of requirements

The input of requirements is delivered by different sources. Managing these sources is an important requirement. In the following subsection the different are described.

Direct enter

The system must be able to receive requirements by direct entering the text in the system.

From Documents

The system must be able to receive requirements from external documents. The documents should be free text or structured text.

Examples:

1. Legal documents
2. Customer specifications
3. Standards
4. Legacy projects
5. Results from issue management

From external Requirement Management tools

The system should be able to read in data from external requirement management tools

Able to capture requirements in the system

The User should be able to capture requirements in the system.

Extracting Requirement based on Ontology

In US2.04 “Ontology based Requirements Formalization and validation” as an improvement of requirements definition is targeted. Therefore a mechanism is needed to derive requirements from Documents based on an ontology.

2.4.1.2 Breakdown of Requirements

According to the “Engineering V” a breakdown from system level to component level should be possible

2.4.1.3 Able to Validate Requirements

The system must support the validation process of requirements. This means that the content of the requirement can be compared to the “as is” state of the implementation of the component or system. Therefore the requirement must be linked to the according components or systems. In general a requirement describes quality properties of a system or quantitative properties of a system. The validation process compares the “to be” state with the “as is” state.

Create relations between requirements and objects in the system

To associate requirements to components or systems, a link between requirement and component is needed. These links are created manually and are used to visualize dependencies and support the validation process. These links have the cardinality n:m. This means one requirement can be associated to multiple components and systems, and one component can have associated multiple requirements.

Validate Qualitative

Qualitative requirements can be subjective. (e.g. is not loud, is smooth, etc.) or objective (e.g. light switches on automatically when it is dark). The validation result is clear yes / no. In the most cases qualitative requirements will be answered manually. The validation result will be stored on the relation/link between Requirement and Component/System.

Part of the requirement definition is to specify the range of validation results. In the case of qualitative results the range is of the type Boolean and may have the value true / false / nil. When the link between Requirement and Component/System is created the result value variable is promoted to the link.

While validation the validation state is set to an appropriate state (true / false) The Validation may be supported by workflows.

Validate Quantitative

Quantitative requirements are measurable e.g. Sound pressure level is less the 20db. Following validations should be supported:

1. The value is equal, value is within a specific range
2. The value is greater than a specified level
3. The value is less than a specified level
4. The value is less equal than a specified level
5. The value is greater equal than a specified level

The validation compares the “As Is Value” with the threshold value. The “As Is Value” is derived by a separate validation process. The validation process is specific for a requirement (eg. determine sound pressure by test, determine maximum displacement by a FEM - Simulation).

The validation result will be stored on the relation/link between Requirement and Component/System.
Part of the requirement definition is to specify the range of validation results. In the case of quantitative results the range is of the type real. When the link between Requirement and Component/System is created the result value variable is promoted to the link.

While validation the validation state is set to an appropriate state (true / false) The Validation may be supported by workflows. This workflow can branch to the needed validation process trigger or direct to the process.

Representation of the validation result

The validation result should be represented on the link between requirement and component/system.

This simplifies the reuse of requirements. The “To Be” Variable is promoted to the link between requirement and component/system. The promoted Variable can be filled with the “As Is” value while the validation process.

Validate on Component level

The requirements on component level are validated, if all linked requirements are satisfied. The “overall validation state” is set to true. The “Overall Validation State” can be defined on component level if the requirements are independent of the use of the component/system. When the component is used in different systems and the component need has to be re-validated than the according “Overall state” is promoted to the link of the component to the higher level system.

Validate on System level

The system validation of requirements need the roll up of the component overall validation states and the system specific requirements.

Manual Validation

If there is no validation process linked to the validation, the validation result can be entered manually by authorized persons.

Automatic Validation

If the validation process is integrated in the system, the process can be triggered from an event (eg. a specific workflow step) The execution of the validation process can be automatic or interactive. Getting the validation values according to the requirements in the system can be automatically happen, when the process is integrated in to the system.

2.4.1.4 Reuse Requirements

Supporting the reuse of requirement is possible when requirements are stored in libraries. From that library the needed requirements can be collected for a new project. To simplify the search of existing requirements in the library, they should be classified.

Requirements library

Once a requirement is defined it should be stored in a library. This enables the reuse of requirements and requirements hierarchies. The library can be structured by a classification schema. This allows to find specific requirements for reuse more easily.

Classification of Requirements

The library can be structured by a classification schema. This allows to find specific requirements for reuse more easily. High level structure of a classification:

1. Domain
    a. Legal requirements
    b. Standards
    c. Certifications

The classification concept helps to identify overlapping requirements form different sources

Arrange existing Requirements to a new project specific set

If a new project is started an early step is to collect the requirements. The collection is normally based on older projects. The requirements are referenced into a project specific top-level node. In the next step the Requirements are reviewed

1. If a requirement fits, it is kept unmodified
2. If a requirement does not fit it is marked as stroked out, but kept in the list for documentation reasons
3. If a requirement fits partially a new release is created which can be modified

2.4.1.5 Maintain life cycle of requirement

Requirements need to be created initially. While requirements may change over the time, e.g. legal requirements, existing requirements should be changeable. To track the modification over the time, the system should support releasing and baselines of requirements. A requirement should be flagged as valid by date dd.mm.yyyy.

Create a new requirement

The system is able to create new requirements by importing text from Word, Excel, PDF, HTML

Based on domain specific ontologies the text can be interpreted and searched for requirement specific information.

The requirement is tagged as a qualitative or quantitative requirement. In a Text field the requirement is entered manually or automatically. If the requirement is a quantitative requirement the threshold values are defined. The threshold is defined by:

1. Name
2. Relation
    a. =, equal
    b. <, less
    c. >, greater
    d. <=, less equal
    e. >=, greater equal
    f. <>, not equal
    g. <<, significant less
    h. >>, significant greater
3. Minimum Value
4. Unit of measure
5. Relation
    a. =, equal
    b. <, less
    c. >, greater
    d. <=, less equal
    e. >=, greater equal
    f. <>, not equal
    g. <<, significant less
    h. >>, significant greater
6. Maximum Value
7. Unit of measure

Modify an existing requirement

The system enables the user to modify existing requirements if specific circumstances occur:

1. The requirement is not already used
2. The requirement is not released

Release requirements

If a requirement is finally defined it should be protected against accidently modifications and flagged for use.

Therefor a state must be applied to the requirement. The state locks the requirement for modifications.

The state controls the visibility of a requirement to specific user groups.

Life cycle of requirements

The following Life cycle states of requirements are available:

1. Work in progress
2. For review
3. Released

4. Legacy
5. Obsolete

Work in Progress indicates that the requirement is still in definition phase. *For review* indicates that the requirement is in review by a specific user group. *Released* indicates that the requirement can be used for requirement specification. *Legacy* indicates that the requirement is valid in old or running project, but should not be used in new projects. *Obsolete* indicated that the requirement is no longer valid, it is only used by already retired project which will no be longer maintained.

2.4.1.6 Configuration and Variants of Requirements

Requirements may be different for different locations (e.g. European, Asian or North American law) or different climatic areas etc. To maintain these different views on the requirements domain specific sets of requirements should be able to be flagged by variant information. This flag will indicate if a requirement is relevant for all cases or only for specific cases, e.g. Europe.

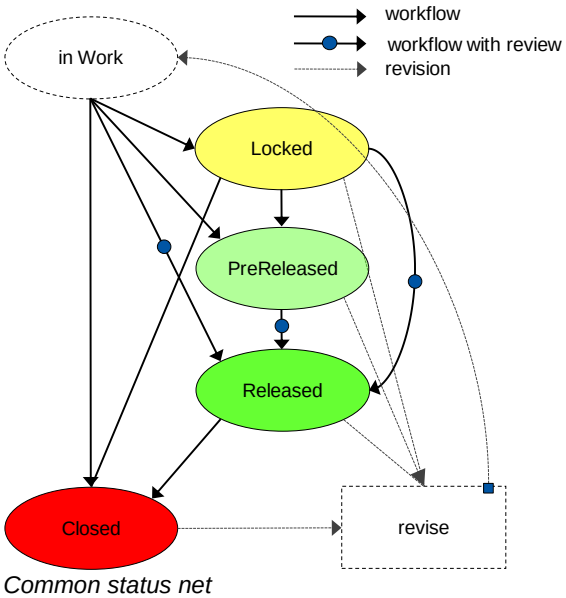
2.4.2 Lifecycle / Workflow Management

The focus of Lifecycle Management includes:

- The definition of release levels of objects during their whole product lifecycle
- Objects status values and possible transitions
- Target objects to which the release statuses are applicable
- General user roles

2.4.2.1 General Status Net

The most important information of the object release status is to identify whether an object revision is valid or not. A release status will be applied to any object on any level of the product structure. The following figure shows the common release statuses and their transitions for most objects.
The status of a revision can be increased without creating a new revision, but decreasing the status is not possible. Revisions with a release status applied cannot be modified anymore to allow traceability of changes in the system. Modifications can be done only by creating a new revision. In general, only the latest revision of an item can be revised.



Common status net

Status “in Work”

- If a new item is created it has no status assigned. This represents the “in Work” in the status network figure.
- The item may be modified (set attributes, create/remove relations, etc...) as long as it has no status.
- It is not allowed to revise items which are “in Work”.
- In order to assign a status, the user has to send the item into a workflow.

Status “Locked”

- The status “Locked” is used to prevent further modifications to a certain item revision. This is useful for example to freeze intermediate work results for communication with other process participants.
- The item revision is protected against further modification.
- This status can only be set for items which are “in Work”.
- Items with status “Locked” can be set to “Released” or “Closed”.
- Items with a status can be revised to create a new "in Work" revision.

Status “Pre-Released”

- The status “Pre-Released” is used to confirm that the item revision is ready to be used for prototype and pre-series production and hand over the intermediate engineering results to the manufacturing department(s) and to communicate the maturity of items in manufacturing.
- The item revision is protected against further modification.
- No review required.
- This status can only be set for items which are “in Work” or “Locked”.
- Items with status “Pre-Released" can be set to “Released” or “Closed”.
- Items with a status can be revised to create a new “in Work” revision.

Status “Released”

- The status “Released” is used to confirm that the object is ready to be used officially for the purpose it is meant to be used.
- The item revision is protected against further modification.
- A review is mandatory.
- Assignment in the solution item folder of a change order is mandatory.

The “Released” status can be set from the initial status “In Work”, “Pre-Released” and the “Locked” status and will need to fulfil object dependent pre-conditions.

Status “Closed”,

- The “Closed” status blocks a single revision to prevent further usage, for example in a BOM or a product.
- Creating new revisions is still possible.
- It will be possible to set the status “Closed” either directly without review or with reviewers included. The decision has to be made by the user that starts the workflow.

2.4.3 Validity, version management , revisions of Artifacts

To ensure traceability and documentation requisites a version management of the specific Artifacts is needed. Any artifact during the product life cycle (element of a model, a requisite, a test, a function, a library etc.) may change (i.e. modified, extended, fixed, etc.) long the live of the product. Thus, it is necessary to be

able to track the modification history of each element. It is important to track those changes, and be able to determine to which extend each modification affects other elements

The system should be able to create structures of artifacts with valid revisions or versions of artifacts.

Open point to be closed later: list of artifacts needs to be provided by IOS

Artifacts must be marked for validity and maturity.

This information needs to be refined during the further steps of the project

2.4.4 Change Management and Traceability of changes

Artifacts in a final state, which are out of Date or need to be modified should run through an
Change Process. This change process must be configurable. The system should deliver standard Change process support

This information needs to be refined during the further steps of the project

2.4.5 Project management and Schedule Management

In order to manage projects the system should support capabilities of a project-management scheduling system.

This information needs to be refined during the further steps of the project

2.4.6 Organizational structure: groups and roles

In order to manage resources the system should provide capabilities to maintain organizational structures with groups and roles.

This information needs to be refined during the further steps of the project

2.4.7 Security and access management

Accessing the data in the system must be limited to authorised persons. Based on the organisational structure and projects the access should be controlled and traced.

This information needs to be refined during the further steps of the project

2.4.8 Data exchange: frequency and format of data exchange

This information needs to be refined during the further steps of the project

2.4.9 BOM-Management

The system should support the management of Bill Of Materials (BOM)

This information needs to be refined during the further steps of the project

2.4.10 Consistency management

The consistency of the created date during different project phases must be managed. Therefore checking of readiness of a deliverable to a due date, data completeness and data validity must be supported

This information needs to be refined during the further steps of the project

2.4.11 Product Variant & Configuration Management

The same system may occur in different variants or configurations. The system must support different variants, e.g. locale specifics.

This information needs to be refined during the further steps of the project

2.4.12 Artifact Management

The system must manage the specified artifacts that are needed for describing the system. Artifacts should be revised, project specific, and access controlled. The creation and maintenance of artifacts should be supported by workflows.

This information needs to be refined during the further steps of the project

2.4.13 Ontology Management

To enable standards of problem descriptions and requirement specifications domain specific ontologies will be created in the subprojects.

Managing the Ontology in the system is required. The creation and modification of the domain specific ontologies must be supported. While defining requirements for a project the system should provide access to the already existing ontologies.

This information needs to be refined during the further steps of the project

2.4.14 Reporting & View Management

This information needs to be refined during the further steps of the project

2.4.15 Documentation Generation

This information needs to be refined during the further steps of the project

2.4.16 View Management

This information needs to be refined during the further steps of the project

2.4.17 Process control

This information needs to be refined during the further steps of the project

3 Terms, Abbreviations and Definitions

This list is a copy taken from the project glossary that is extended by WP 608 specific terms and definitions. This might be removed in further versions of D608.011 or the additional information will be added to the AVL maintained project glossary. It is only kept here for WP 608 reference purposes.

CRYSTAL	**CR**itical S**YST**em Engineering **A**cce**L**eration	
R	Report	
P	Prototype	
D	Demonstrator	
O	Other	
PU	Public	
PP	Restricted to other program participants (including the JU).	
RE	Restricted to a group specified by the consortium (including the JU).	
CO	Confidential, only for members of the consortium (including the JU).	
WP	Work Package	
SP	Subproject	
ADL	Architecture Description Language	
AGATHA	Atelier de Génération Automatique de Tests Holistiques pour Automates	Tool
AHRS	Attitutde Heading Refernce System	
AIPP	ARTEMIS Innovation Pilot Programmes	
ALM	Application-Lifecycle-Management	
ARAMiS	Automotive, Railway and Avionics Multicore Systems	Project
ASAM	Association for Standardization of Automation and Measuring Systems	
ASP	Artemis Sub-Programmes	
ATAM	architecture tradeoff analysis method	Method
ATESST	Advancing Traffic Effciency and Safety through Software Technology	Project
ATM	Air Traffic Management	
AUTOSAR	AUTomotive Open System Architecture	Method
Bugzilla	Bug tracking system	Tool

CAGR	Compound Annual Growth Rate	
CCA	Common Cause Analysis	Method
CCC	Completeness, Consistency, Correctness	
CESAR	Cost-efficient methods and processes for safety relevant embedded systems	Project
COTS	Commercial/Components-Off-The-Shelf	
CRTP	Cooperative Reference Technology Platform	
CRYSTAL	CRitical sYSTem engineering AcceLeration	Project
DASIA	Data Systems In Aerospace	
DL	Dissemination Leader	
DO	Domain Ontology	
DODT	Domain Ontology Design Tool	
ECC	Error-correcting Codes	
ECS	Environmental Control System	
ECU	Electronic Control Unit	
EDL	Engineering Domain Leader	
EDP	Engineering Domain Partner	
EDTR	Engineering Domain Technical Representive	
E-FCS	Electric Flight Control System	
EICOSE	European Institute for Complex Safety Critical Systems Engineering	
EL	Exploitation Leader	
EM	Engineering Methods	
ERTMS	European Rail Traffic Management System	
EUCAR	European Council for Automotive R&D	
FME(C)A	Failure Mode Effect (and Critically) Analysis	
FMU	Functional Module Unit	
FPGA	Field Programmable Gate Array	
FTA	Fault Tree Analysis	
GA	General Assembly	
GI	Gesellschaft für Informatik	
GNSS	Global Navigation Satellite System	

HIL	Hardware-In-The-Loop	
HW	Hardware	
iFEST	industrial Framework for Embedded Systems Tools	Project
IMA	Integrated Modular Avionics	
INCOSE	International Council on Systems Engineering	Organisation
IOS	interoperability standards	
IPL	IOS Platform Builder Leader	
IPR	Intellectual Property Rights	
IRS	Inertial Refernce System	
ISL	IOS Standardisation Leader ARTEMIS-2012-1 Full Project Proposal addressing AIPP CRYSTAL	
ISP	Infrastructure Service Provider	
ISVV	Independent Software Verification and Validation	
MARTE	Modeling and Analysis of Real-time Embedded Systems	Project
MBAT	Model-based Analysis and Testing	Project
MBSE	Modell based systems engineering	Method
Metrino	an integrated set of tools to support the validation and quality assurance of models based on OCL	Tool
MIL	Middleware-In-The-Loop	
ModelBus	model-driven tool integration framework	Tool
MSE	Mission Support Equipment	
NFC	National Coordinator for Funding	
OCL	Object Constraint Language	Method
OEM	Original Equipment Manufacturer	
OMG	Object Management Group	Organisation
OSATE	open-source AADL tool environment	Tool
OSB	Operative Steering Board	
OSLC	Open Services for Lifecycle Collaboration	
PCB	Printed Circuit Board	
PGA	Programmable Gate Array	
PLCS	Product-Life Cycle Support	

PLE	Product Line Engineering	
PLM	Product-Lifecycle-Management	
PM	Project Manager	
PQA	Project Administrator	
R&D	Research & Development	
R&T	Research & Technology	
RECOMP	Reduced Certification Costs Using Trusted Multi-core Platforms	Project
ReqIF	Requirement Interchange Format	
RQ	Requirement	
RTE	Run-Time Environment	
RTL	R&T Leader	
RTP	Reference Technology Platform	
SAFE	Safe Automotive soFtware architEcture	Project
SafeCer	Safety Certification of Software-Intensive Systems with Reusable Components	
SCADE	a model-based development environment dedicated to critical embedded software	Tool
SEE	System Engineering Environment	
SIL	Software-In-The-Loop	
SLM	System-Lifecycle-Managment	
SME	Small and medium enterprises	
SOA	Service Oriented Architecture	
SP	sub project	
SP	Sub-Project	
SSB	Strategic Steering Board	
SUT	System Under Test	
SW	Software	
SysML	Systems Modeling Language	Method
TCG	Test Case Generation	
TFP	Tool Function Provider	
TIP	Tool Infrastructure Provider	
TRL	technology readiness level / technology maturity level	

UC	Use Case	
UML	Unified Modeling Language	
US	user storry	
V&V	Validation and Verfication	
W3C	World Wide Web Consortium	
WCET	Worst-Case Execution Time	
WP	Work Package	

Table 3-1: Terms, Abbreviations and Definitions

4 References

[IOS Workshop] Presentation of IOS Workshop WP 2.08: Public Use Case Aerospace Draft Proposal for IOS Workshop

[CRYSTAL Proposal] CRYSTAL Full Proposal Part B