ITEA

Information Technology for European Advancement

Task 1.8 Requirements Management method definition (D.1.8.2)

Version 01 - Public

Edited by Loek Bokhorst

Software Development Process for Real-Time Embedded Software Systems (DESS)

ITEA COMPETENCES involved:
1) Complex Systems Engineering
2) Communications
3) Distributed Information and services

1. LIST OF TABLES.

2. LIST OF FIGURES.

3. INTRODUCTION.

The purpose of Requirements Management is to establish a common understanding between the customer and the software project of the customer's requirements that will be addressed by the software project.

Requirements Management involves establishing and maintaining an agreement with the customer on the requirements for the software project. This agreement is referred to as the "system requirements allocated to the software." The "customer" may be interpreted as the system-engineering group, the marketing group, another internal organisation, or an external customer. The agreement covers both the technical and non-technical (e.g., delivery dates) requirements. The agreement forms the basis for estimating, planning, performing, and tracking the software project's activities throughout the software life cycle.

Analysis and allocation of the system requirements is not the responsibility of the software engineering group, but is a prerequisite for their work.

The allocation of the system requirements to software, hardware, and other system components (e.g., humans) may be performed by a group external to the software engineering group (e.g., the system engineering group), and the software engineering group may have no direct control of this allocation. Within the constraints of the project, the software-engineering group takes appropriate steps to ensure that the system requirements allocated to software, which they are responsible for addressing, are documented and controlled.
To achieve this control, the software-engineering group reviews the initial and revised system requirements allocated to software to resolve issues before they are incorporated into the software project. Whenever the system requirements allocated to software are changed, the affected software plans, work products, and activities are adjusted to remain consistent with the updated requirements.

3.1 Purpose and scope

This document describes the ITEA DESS Requirements Management method.
The scope of the requirements management method is not restricted to software development but it also covers the activities related to the system/product level development.
Requirements management has a strong relation to other activities like requirements engineering, software engineering, configuration- and change-management, quality assurance. These activities are not covered here.

4. OVERVIEW.

4.1 The requirements management method.

Requirements Management and Requirements Engineering is the process of systematically eliciting, organising, and documenting the requirements for a complex system, such that the agreement between the customer and the product development is established and maintained.

The structuring and administration of the requirements related information, which is processed during the acquisition, derivation, analysis, verification, validation is done in order to be able to give access to the status, history and relations of the parts/components of a product.

Traceability of requirements over the entire product lifecycle enables the product development team members to perform impact analysis. E.g. the project leader can inspect if the requirements in the functional requirements specification cover the requirements written in a commercial requirements document. The system architect can inspect the effect of changing a specific resource constraint.
Each requirement must be labelled with a unique identification so one can to refer to this requirement.

In the early stage of the product development a requirements management plan is created. This plan describes the organisation of the activity in terms of the departments/persons related to the process, responsibilities, the information that will be managed and made traceable, the effort to be spent, the tailoring of the requirements management method, the requirements management tool aspects etc. Refer to: The Requirements Management Plan on page 18.
The information is managed is defined by listing the attributes, their meaning, their values, the owner. Refer to Requirement Attributes on page 43.
The component information model supports the management of the component development. It contains a detailed status of the component on quality requirements and achievements, architectural and -design information. Refer to: The Requirements Management Information Models on page 31

4.2 Document overview.

Table 1 Document overview.

Chapter	Contents
The scope of the Requirements Management method.	Describes the context of the requirements management activity. The organisational interfaces, the different types of product development are explained.
RM activities in the process.	Describes in global terms, what kinds of requirements management related activities are done in the phases of the development process. Project management, requirements approval and Validation and verification subjects are covered.
Project management activities related to requirements management	In more detail the project management actions in relation with requirement management are described. The requirements management plan contents is explained. An example of the project characteristics is given. The size and type of a project influences strongly the emphasis on requirements management to be put. See: On page 25 and 27
Requirements management information models	Requirements management is the management of information. This chapter describes how this information is structured. An example of a component information model is given. At the component level quite some information is to be available. Like the status of the development, the status of the Validation and Verification, analysis information, architectural models, design information etc.
Traceability of requirements information.	When requirements are traceable they are useful. Requirements are to be traceable to useful information and to the more detailed descriptions of it, like use cases, test results. A number of techniques to implement this are described.
Classification	An overview of the type of requirements is given. The classification of requirements is given in terms of functionality, data, and non-functionality. Especially the non-functional requirements types are presented. See: Mastering the Requirements Process by Suzanne Robertson, James Robertson
Requirements Identification	This chapter presents the syntax and sementics of a requirement identification. This might be usefull for the manual technique as well.
Requirement Attributes	Attributes are information items, which have a relation to identified requirements or other data like design model, test cases. In this chapter examples are given of the attributes can be defined for the user requirements, test cases etc. Refer to The Requirements Management Plan on page 18
APPENDICES.	The appendix contains an example of a checklist, which can be used by a reviewer of the requirements.
Standards.	This chapter gives an overview of the “world” standards, to which this document should comply.

Chapter	Contents
Requirement Guidelines	In this chapter the definition is given of the criteria of a correct requirement.

4.3 Glossary

4.3.1 Definitions.

Table 2 Glossary: Definitions.

Definition	Description
Artifact	A requirement or any other document containing one or more traceability items.
attribute	For each traceability item you have identified, list what attributes you will be using and briefly explain what they mean. For example, the following attributes might be specified for a traceability item of “feature”: status, priority
Attribute matrix	Per requirement type a list of related attributes is shown, depending of the attribute-list view defined.
Impact analysis	Impact analysis is the inspection of the relations of a traceability item.
Traceability	Traceability of the requirements of the CRS to the requirements of the FRS etc. means that a link has been created between these.
traceability item	A traceability item is any project element that needs to be explicitly traced from another textual or model item in order to keep track of the dependencies between them. With respect to Rational Requisite Pro, this definition can be rephrased as: any project element represented within RequisitePro by an instance of a RequisitePro requirement type.)
Traceability Matrix	A matrix of two < traceability item >’s, which shows the defined links between them
CAD support	Computer Aided Design support department of SLE

Table 3 Glossary of artefact names.

Artefacts	Description
[1] ADI	Architecture Definition Interface specification
[2] Ats	Acceptance test specification
[3] Atr	Acceptance test result
[4] HWs	HardWare specification docs.
[5] Itr	Integration test result
[6] Its	Integration test specification
[7] RMP	Requirements Management Plan
[8] Str	System test result.
[9] Sts	System test specification.
[10] SWs	SW specification docs.
[11] SYs	SYstem specification docs
[12] SYt	SYstem test specification docs
[13] UD	Unit Design doc.

Artefacts	Description
[14] Utr	Unit test cases result
[15] Uts	Unit test cases specification
[16] UR	User Requirements doc.

Table 4 Glossary: Terms.

Term	Description
CCB	Change Control Board
CMP	Configuration Management Plan
CR	Change Request
FRS	Functional Requirements Specification
HSI	Hardware Software Interface
HW	Hardware
IC	Integrated Circuit
SCP	Software Creation Process.
SPMP	Software Project Management Plan

4.4 References

Table 5 References

Title, author, supplier, date and status.	Reference
[1] "Requirements Specification template. Draft V 01- 2000-11-22"	***ITEA DESS***
[2] "Requirements Specification template description, Draft V 01- 2000-11-22"	***ITEA DESS***
[3] Software Development Process for real-time Embedded Software Systems(DESS)	***ITEA DESS***
[4] WP1.3 Study of timing, memory and other resource constraints	***ITEA DESS***
[5] Task 1.4 – Definition of Components and Notation for Components (D.1.4.2) *October 2000*	***ITEA DESS***
[6] Recommended Practice for Software Requirements specification, Institute of Electrical and Electronics Engineers, 1984.	IEEE Standard 830-1998,
[7] Mastering the Requirements Process by Suzanne Robertson, James Robertson	Amazon.com http://www.systemsguild.com/GuildSite/Guild/Books.html
[8] Software Requirements. Practical techniques for gathering and managing requirements throughout the product development cycle by Karl E. Wiegers. Released: 09/29/1999	Microsoft Press. www.Mspress.microsoft.com
[9] Systems engineering coping with complexity by Richard Stevens, Peter Brooks, Ken Jackson and Stuart Arnold, Prentence Hall Europe 1998.	ISBN 01309050858
[10] Managing Software Requirements A Unified Approach by Dean Leffingwell, Don Widrig, Addson Wesley, April 2000	ISBN 0201615932
[11] Programme Management and System Engineering Capability Maturity Model. , 25 September 1998.	TTM/DLS/98/5332/V0.2
[12] IEEE Standard Glossary of Software Engineering Terminology, Institute of Electrical and Electronics Engineers, 1990.	IEEE Standard 610.12-1990,
[13] Software Configuration management. By Ronald Berlack, By John Wiley & Sons, Inc., 1992	ISBN 0-471-53049-2
[14] http://www.esi.es/Publications/Articles/PDF/99rtpl.pdf Requirements Traceability for Product-Lines	

5. THE SCOPE OF THE REQUIREMENTS MANAGEMENT METHOD.

Requirements management is part of all phases of the entire system life cycle, ranging from market analysis and top-level customer requirements to system development and maintenance, final shutdown, and removal.
In system and software development different requirements engineering methodologies are used, but the necessary requirements management activities are nearly identical. Therefore, the requirements management method defined here is valid for any application domain.

5.1 Organisations involved in requirements management.

5.1.1 Introduction.

One can distinguish different levels of requirements. The [16] UR (User requirements doc) is in most cases the highest level of specification, the system requirements specification is a lower level, because the contents is based (controlled), on the [16] UR. The role of the author of the [16] UR acts as the customer of the author of the system requirements specification. The authors may be part of different departments. The following will explain the organisational relations between the responsible employees at the different specification levels of the requirements.

5.1.2 Organisational aspects.

The organisational relations between the responsible employees at the different specification levels of the requirements are important. The spectrum extends from working groups within an organisational unit through different departments of a development group to a customer-supplier-relationship.
For small projects/organisations, functions such as test management and quality assurance may be performed the project team. For larger projects/organisations, these functions may be performed by independent groups.

Table 6 Stakeholders and their contributions to the requirements

Organisation	Performs
Marketing	Specifies high level requirements and request changes; Provides business requirements.
Commercial Department	Specifies commercial requirements and request changes. Specify business, functional, and performance needs; request changes.
Account Management,	Is the formal contact person to the customer.
Users	Describe the user requirements and quality attributes; review the requirements; request changes
Test Management	Reviews the specifications, write the test cases and executes the tests
Configuration management	Writes the configuration plan and performs the base lining of all artefacts of the project. The CCB (Change Control Board) is defined, which decides on the acceptance of Change requests and problem reports.
Quality department, Process Improvement	Defines and checks the development process. Checks the application domain world standards.
Hardware Engineering	Specify the hardware software interface and characteristics, build the hardware platform of the product.
Systems Architecture	Allocates the system requirements to software; Defines the system architectural design and hardware software interface
Requirements management	Database administration, responsible for the consistency of the requirements traceability.
Methods and Software Development design flow tooling	Defines the Software Design Environment, standards
Legal Department	Handles licensing of tools and components.
Project management	Product level project management.
Senior Management,	Verifies the process defined for the overall project and takes corrective actions when problems arise.

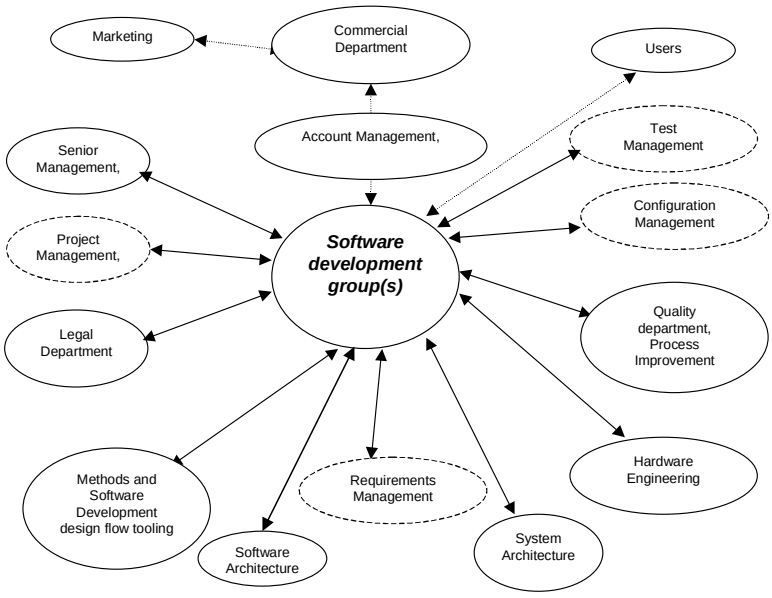

Figure 1 Roles in requirements management.

Legend: Depending on the size of the development group, some functions may be performed by an independent team. These functions are indicated by the dotted line

5.1.3 Product-line development.

A product line is a group of related products/components that, taken together, address a specific market segment. Product lines are most efficiently built from a common set of reusable assets, such as domain models, requirements, architectures, software components, test plans, project schedules, tools, and environments.

It is important that one organisation is responsible for the reusable components at the product-line level. The product-line architects are responsible for the product-line architecture and approve the architecture of the components under development at the product/component level.
The product-line program manager defines the roadmap of the product-line. The person initiates projects for new components, which are not directly coupled yet to a product-level development. The product-line project management manage all product-line-related projects.

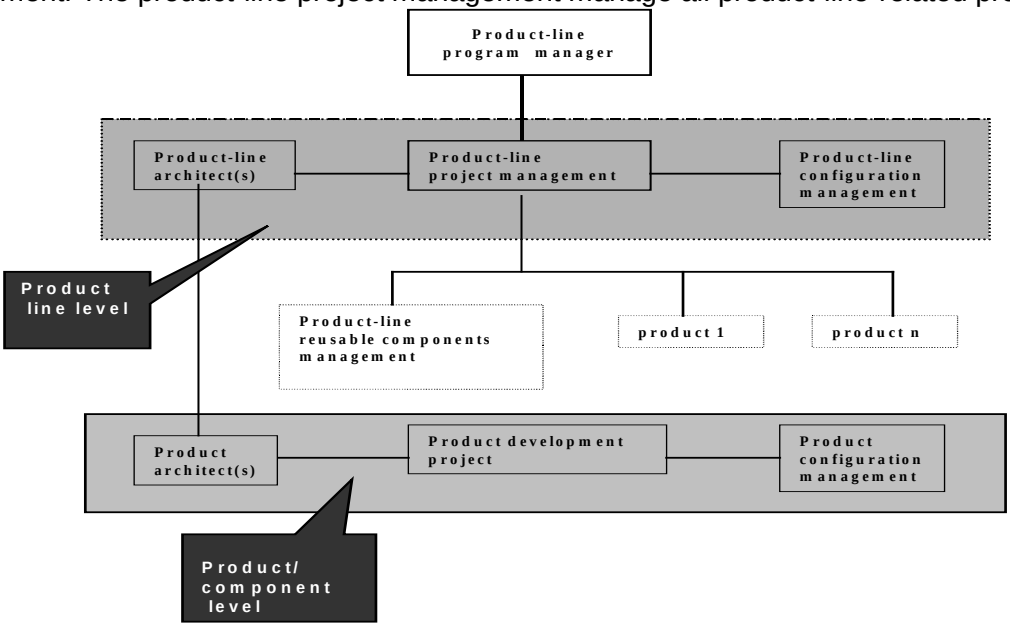

Figure 2 Product line s.w. development.

5.2 Product-line related requirements

A product-line is built on the reusable components.

Reusable software must be supported by reuse of requirements. This means that a central repository of requirements that can be configured in different ways for different products is a necessity.

When a candidate component consists of one or more reusable sub-components, the sub-components are managed as independent units in the same way a component is managed.

For the product-line one or more “collections” of requirements are defined, depending on the type of product.
Examples:
1. The requirements for a specific region for a specific product range

2. The requirements for a specified representative of the product line structure (for a given region, or for all regions)
3. The common requirements for all members of a product line
4. The common requirements and all variations that are to be supported for a specific feature area.

When a component is re-used in different products then some changes to the component can influence the performance of the product. The impact of a change has to be discussed with the group responsible for the products.

5.2.1 Multi-discipline product development.

The following picture shows the software created in a hybrid development.
In this type of development HW/SW co-design is done.

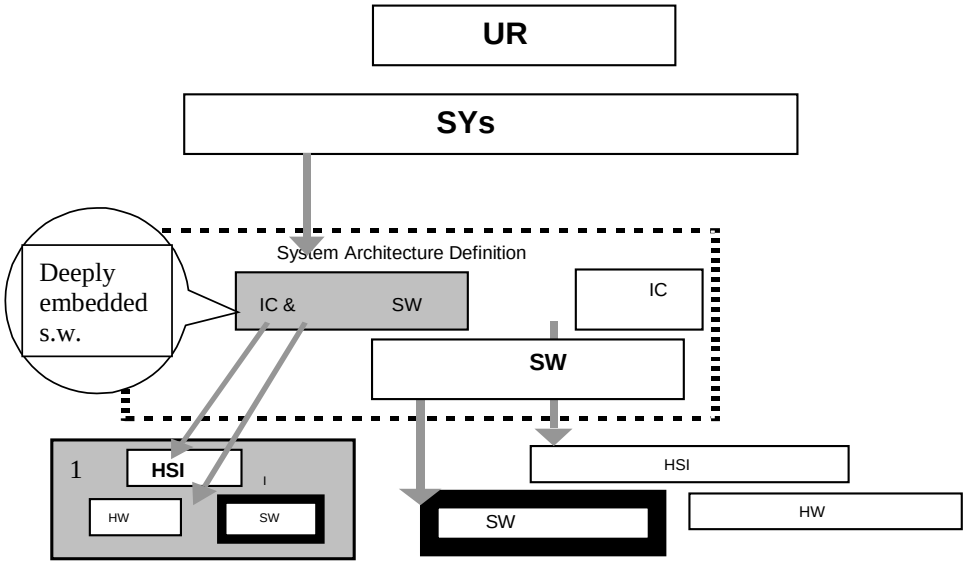

Figure 3 Overall System development

The software to be developed may be part of an IC (is deeply embedded) and/or form a component with one or more IC's.

RM ACTIVITIES IN THE PROCESS.

6.1 The Inception Phase.

The following Requirements engineering tasks are performed:

1. The analyst collects the requirements of a project from the customer. These requirements could be in the form of a [16] UR , any other document, by E-mail or by personal communication with the client. The personal communication has to be **documented and approved** by the customer before it can be incorporated in the requirements document.

   At the end, a list is prepared of documents received and the support documents with their version numbers.

2. It is determined what the broad constraints for the project are, what exactly the client is looking for and in what time frame the product must be built.

3. It is identified what the target environment, development environment and tools required for development are, including target systems, test equipment, software - source, compilers etc., including version numbers.

4. People are identified who should be contacted for different changes and for clarification.

5. The Clarification Request Form is used to get clarification if needed, on various inputs received from the customer.

6. The status of the Clarification Requests is tracked.

Note 1: Points 5 and 6 are suggested strongly when many clarification requests are submitted. This is to ensure that no clarification request is left out and that a periodic auditing of the status of clarification requests is possible. This is similar to Change Request and Problem Report procedures and tooling.
Note 2: In the section above, the contents of items 4 to 6 have to be captured but the mechanism of executing it can be project specific and is specified in the SPMP.
Inception Phase.

In the first phase of a product definition all raw requirements are collected.
These requirements are related to the marketing product requirements. These are the main features of the product, the budget and price of the product and the main milestones.
The result of this phase is the system definition in which the features are made concrete.
The vision of all stakeholders, the product user requirements, the technology requirements etc. are documented in the [16] UR.

With the customer is made an agreement on who is responsible for keeping the UR up-to-date.
The document is maintained throughout the development life cycle either by the customer or by the development project.
The requirements consist of a stable part and of a variable part. Part of the analysis activity consists of analysing the variations in connection with future extensions or product families.
The customer formally has to approve the UR.

The global product requirements are defined:

1. User interface requirements.
2. Technical product context and interfaces. The architecture, communication.
3. Commercial- and market requirements.
4. Product price, development budget.
5. Documentation.
6. The releases of the product.
7. The definition of a product line.
8. Product and project constraints, quality criteria
9. The budget and the development costs.
10. The identification of the new and the re-usable components.

Definition of the features of the product:

1. What are the basic features and extra features.
2. The importance of the features.
3. The risks and costs of the features.
4. The constraints,
5. The quality criteria.
6. The technology to be used.
7. The applicable world standards.

Activities during this phase are:

Project management related activities.

1. The initial project plan is written.
2. A requirements management plan is written, see: The Requirements Management Plan (page 18). This will describe the actual requirements on traceability, which requirement information (attributes) is to be maintained etc.
3. A configuration management plan is written for the change management and the document quality aspects, see: (page 22).
4. The organisations that are involved, the project- and design- meetings, the responsibilities and reporting are defined.
5. All relevant stakeholders formally approve the project plan.

Requirements related activity.

6. The UR is finalised and formally approved. The customer keeps this document up-to-date. If not the development project creates a version of the UR document. The customer and development project team formally approves changes of the UR.
7. A requirements database is created. The attributes, requirement types, users and privileges as defined in the requirements management plans are defined.

6.2 System elaboration phase.

Write the system functional requirements specification
The requirements of the [16] UR are traced to the functional requirements specification.
When changes to the [16] UR are to be made, the change request procedure as defined in the CMP is followed.
Write the system design description. This results in an allocation of the requirements to software, to hardware and to the HSI specification.
From this moment the overall project manager manages the applicable H.W./S.W. sub-projects. The proper execution of the requirements management plan by the sub-projects is verified.

6.3 V&V activities related to requirements management.

Documents produced during the software development process are reviewed or cross checked according to the SPMP. The V&V group is reviewing the artefacts, starting with the UR.

Part of the review process is a check of the fit criteria, the traceability to the respective test specifications, the inspection of the requirements attributes, updating the V&V specifically defined attributes.

The information of the test results is traced to the respective test cases.

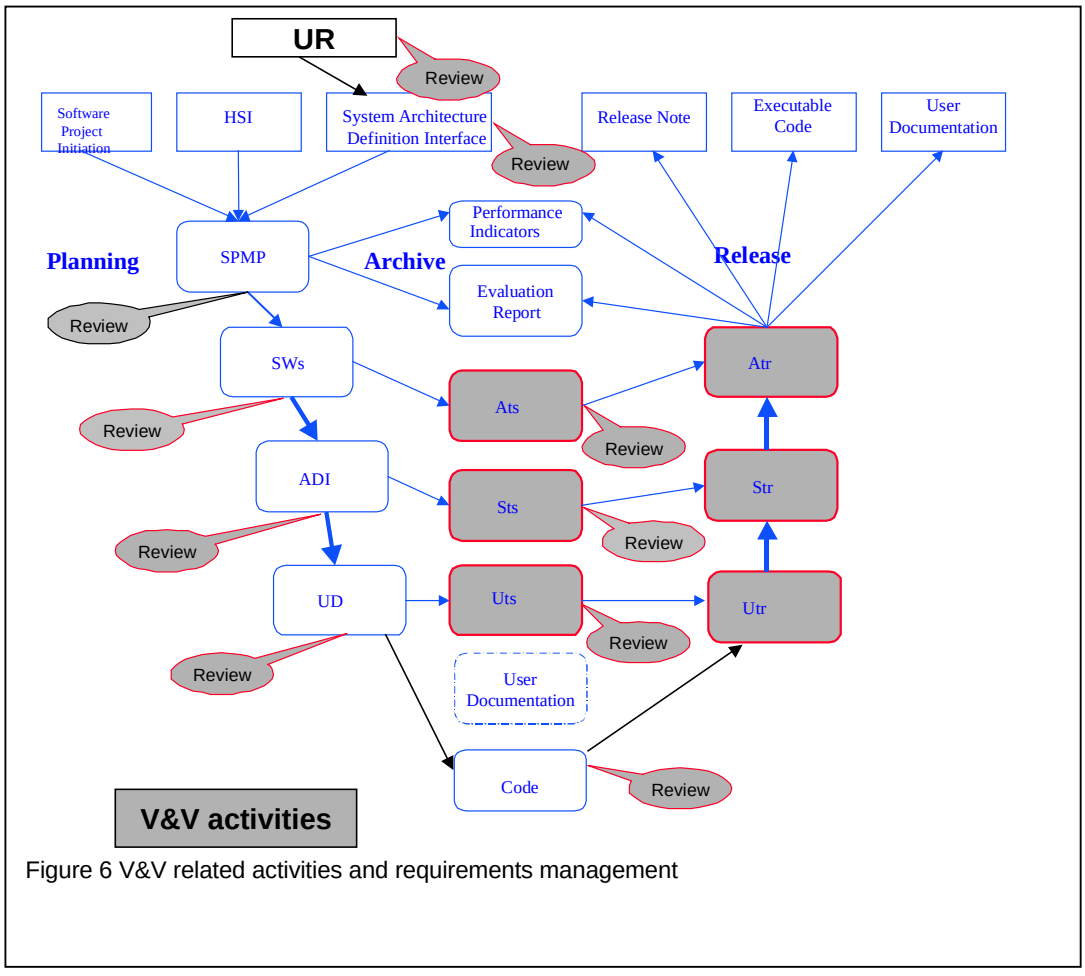

Figure 6 V&V related activities and requirements management

7. PROJECT MANAGEMENT ACTIVITIES RELATED TO REQUIREMENTS MANAGEMENT

The development project manager makes an agreement with the customer about the process of the requirements gathering and approval and change management. The project management plan contains the requirements management activities, the approach in terms of the collection of metrics, (senior management) reporting and effort etc.

7.1 Project Management Plan

The Project Management Plan (PMP) contains amongst other sections the Requirements Management Plan, the Quality Assurance Plan and the Configuration Management Plan.
The PMP is formally inspected and signed by the parties involved.
Adequate resources and funding are provided for managing the allocated requirements.

7.2 The Requirements Management Plan

This section of the PMP describes all requirements management related issues specific for the project. The activities performed depend on the type and size of the project.

7.2.1 Human and non-human resources needed for the Requirements Management activity

The following must be planned or established:

1. The effort and co-operation of the departments / disciplines involved. This includes test management, hardware development, quality department etc.
2. The effort for gathering, review and maintenance activities of the [16] UR, SRS etc.
3. The maintenance and the impact analysis activity.
4. The maintenance and the review activity of the attribute matrices.
5. The metrics which are to be applied.

7.2.2 Management reporting.

What is to be reported and the frequency of reporting are established.
This Software Project Progress Report is distributed to senior management and the overall (system) project leader/manager.
The section 'Requirements Management' of the report contains:

1. Conflicts and issues not resolvable at lower levels are addressed.
2. Action point list. Containing <week nr> <sequence nr>, who, what, deadline, status
3. A summary of the Measurement and analysis information like:
   a. *The status of each of the Major allocated requirements and the planning, costs*
   b. *The number of Change Requests to the allocated requirements including problems encountered since the last report.*
   c. *The cumulative number of changes to the allocated requirements, including total number of changes with the status: proposed, open, approved, and incorporated into the system baseline.*
4. The effort spent on requirements management.

7.2.3 *Required* Tooling.

Specifies the project or organisation RM tool selected.

7.2.4 Information models used

- Document traceability structure.
  It shows all the documents and their relations See Document traceability structure. On page 30.
- Component attribute information model
  The component attribute information model defines the attribute groups related to components, see: (page 34).

7.2.5 Requirements management related responsibilities.

The following RAEW tables define the roles and responsibility related to a number of RM activities.

Table 7 Legend of the RAEW tables.

	Description
R	Responsibility for the proper execution of the activity.
A	Authority to decide on the Go/No Go matters of the activity
E	Expertise on the activity
W	Performing the actual work

Table 8 RM responsibilities: RM maintenance

Activity	Architect	V&V-team	Project leader	RM DB Tool-smith	RM DB Admin.	
Entering requirements UR	RW		AE			
Entering requirements SYs	RAEW					
Entering requirements ADI	RAEW					
Entering requirements SWs	RAEW					
Traceability						
Impact Analysis requirements	W	W	ERA			

Table 9 RM responsibilities: RM types, attributes

Activity	Architect	V&V-team	Project leader	RM DB Tool-smith	RM DB Admin.	
The definition of the attributes of the requirements types UR	REW		A		W	
The definition of the attributes of the requirements types SYs	REW		A		W	

Activity	Architect	V&V-team	Project leader	RM DB Tool-smith	RM DB Admin.	
The definition of the attributes of the requirements types SAD	REW		A		W	
The definition of the attributes of the requirements types SWs	REW		A		W	

Table 10 RM responsibilities: RM tool customization, configuration

Activity	Architect	V&V-team	Project leader	RM DB Tool-smith	RM DB Admin.	
The definition of the users, their privileges etc			RA		EW	
the definition of requirement-attribute views, queries	E	E	RA		W	
Making Backups, base lining docs.			A		REW	
Collecting metrics			RA		EW	

7.2.6 Predefined Views

Different views on process activities must be provided for the different requirement management roles. A view presents only a subset of the existing information to the user. E.g. a project manager is interested in other information details than a V&V expert. A requirements specification document is a particular representation of a view on the existing information related to requirements.

7.2.7 User privileges.

Process and security aspects require that access to particular information must be prohibited for some users. E.g. V&V staff should have only read access to requirements; cost information must be readable only for project managers.

7.2.8 Project tracking activity.

For the requirements management activity in the project a number of sub-activities are defined. Effort spent per sub-activity is reported to the project manager.

Label	Activity	Description of time spent	Details / Remarks
KPA Related			
RM	RMP	Time spent on the RMP	Creation, review, maintenance
	RM support	Support on method and tooling.	Training, documentation, problem solving
	administration	The administrative work related to the RM database	Base lining, user definitions, backups, and configuration of the project.
	Requirements	The work involved with the Insertion of requirements text in the RM database	The setting of the attribute values, importation of requirements documents,
	Change Analysis	Impact analysis of Change Requests	Only analyses. Implementing a CR is part of other activities, e.g. IMP, V&V.
	Communication	Meetings and other communication with customers about the requirements of the project CCB meetings are NOT part of this but are part of Configuration Management	Especially discussion with the customer, sponsor or end-user. Also participation in system architecture meetings needed to get the allocated requirements for the project clear.
	Documenting	All activities needed to get the requirements for the project (including schedule requirements) documented. This is NOT the effort needed to create the SRS.	Effort spent on e.g. [16] UR, FRS, FTS or Project Initiation, including review and rework of the author. Creation/maintenance of the Attribute/Traceability Matrices.

7.2.9 Metrics.

Measurement: Percentage of changed requirements per requirement type

Introduction: The percentage of changed requirements per requirement type gives an indication in the stability of the requirements. Since the development team and the testteam each maintain their own set of requirements, each team measures on the requirements that are allocated to that team.

The development team delivers requirements measures, as given in **Error! Reference source not found.**, the testteam as given in **Error! Reference source not found.**.

If the data is collected regularily, trend graphs can be made, reflecting the stability of requirements over time (see section).

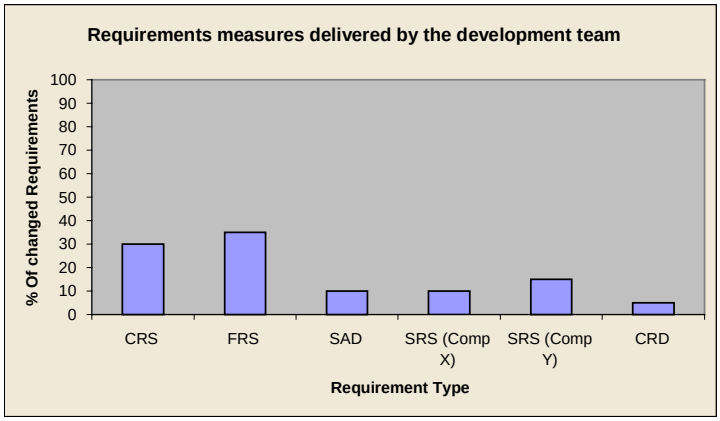

Figure 8 Example: Requirements measures delivered by the development team.

Rationale: The percentage of changed requirements per requirement type gives an indication in the stability of the requirements.
A changed requirement is defined here as: a requirement from which the text is changed. (Requisite Pro supports also monitoring of other types of changes. These are not taken into account at the moment, see section **Error! Reference source not found.**).

Time to collect : Once per week

Required data :

Requirement Type	Responsible for collecting data	1	2	3	4	..
UR	Development team					
SYs	Development team					
CR	Development team					

Table 11 Collecting the percentage of changed requirements per requirement type and per period

Measurement: Trend graphs

Introduction: Trend graphs with the percentage of changed requirements per requirement type gives an indication in the stability of the requirements over time.

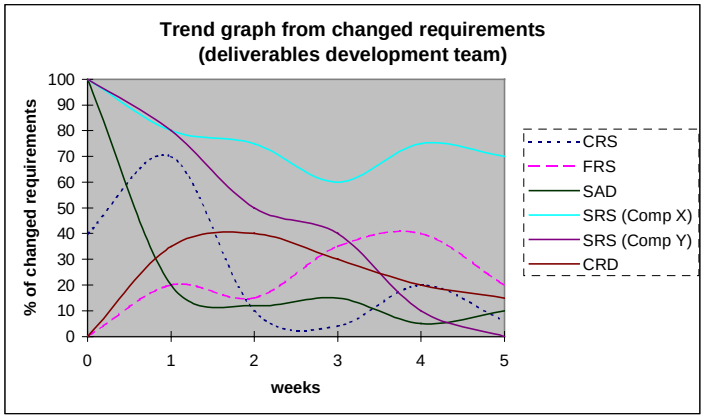

Figure 9 Trend graph from changed requirements from deliverables of the development team.

Rationale: The trend graphs of changed requirements per requirement type give an indication in the stability of the requirements over time.

A changed requirement is defined here as: a requirement from which the text is changed.
Time to collect : The data is gathered each week. The trend graph is made each month.

Required data : Same data as described in section 7.2.9

7.3 Quality Assurance Plan.

The quality assurance audits are dealing with process; product en general and company quality standards related items.
In order to verify the adherence to the procedures defined in the SCP, a Software Quality Officer (SQAO) is assigned to each project. The SQAO performs Quality Audits and reports the process compliance to the Project Manager.

Product domain specific quality.
The focus is the compliance of the specifications to the world standards defined for the domains of the product.

***Safety relate*d.**

For requirements related to safety, a traceability matrix is required. Examples are medical products like a MRI scanner and radiotherapy systems. The correct implementation of safety standards has to be proven, depending on the law (USA).

7.3.1 Change management.

The change management is addressed in the Configuration Management Plan.

The configuration management plan, which is formally approved, will define the level of formalism per artefact.
Less frequent. Depending on the project needs, the frequency of project, configuration management status reporting is documented in the management plan.

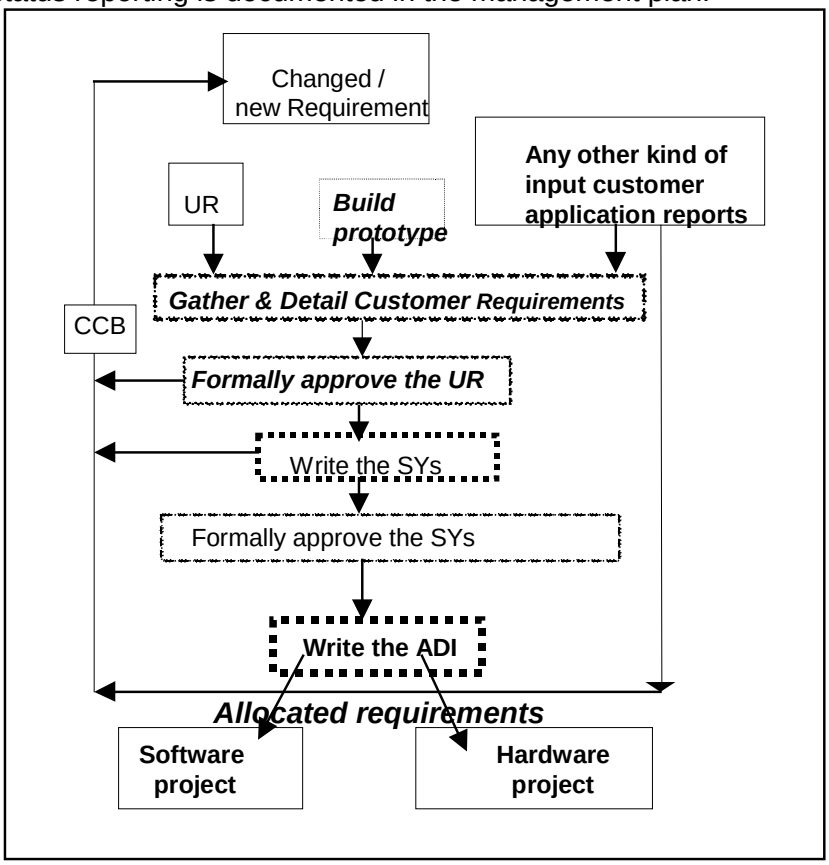

Figure 10 Change management.

7.4 Project characteristics.

A project is an activity performed by one or more persons, in order to reach a defined goal. A customer provides the funding for the project. The customer may be external or internal to the project organisation.
A project can be allocated to a large or small group of persons. A project can be sub-divided into one or more sub-projects. A sub-project can be external or internal to the main project.
The goal of the project can be the delivery of a complete system, a sub-system, or a component. Per deliverable, particular quality factors can be assigned.
The formal way of handling the requirements specifications (at the commercial, system, and /or on component level) depends on the project characteristics.
The project management plan should clearly define how the Requirement Management is organised and planned. A separate section of the project management plan is advised.
Two aspects have a particularly large impact on the project's defined software process:

1. The project type .The project type can either be 'business' or 'economy', see Table 12 Project Catagories.
2. The project size. The pProject size e.g. can be small, medium or large.

This tailoring procedure directly influences the project standards for of the deliverables. This tailoring procedure is described in detail in the SCP.

With this, the following quality level in relation to project tailoring can be defined as in Figure 11.

Table 12 Project Catagories.

Attribute	Category	Characteristics
Size	Small	<= 3 person months net effort
	Medium	>3 person months net effort, <= 12 person months net effort
	Large	> 12 person months net effort. Large projects may be structured in sub-projects or teams.
Type	Business	The requested deliverable is: • To be used by the customer • To be produced in quantities. • *The development results directly, or indirectly, in a product to the end-user market*. This applies to development activities like migration to new HW or SW platforms, problem solving/change requests solving. The development should be according to the highest standards of the creation process defined. A full requirements management plan section is written.
	Economy	The goal of the development is technology development, pre-concept development, demonstration software, and test-software. The development does not results directly, or indirectly, in a product to the end-user market The deliverable is for internal use only. The development should be according to the highest standards of the creation process defined, but the activities are done less formal. Activities, which are performed formally, as defined for the Business category of projects, may be done informally. The Requirements management plan clarifies which activities, and to what extend, is performed.

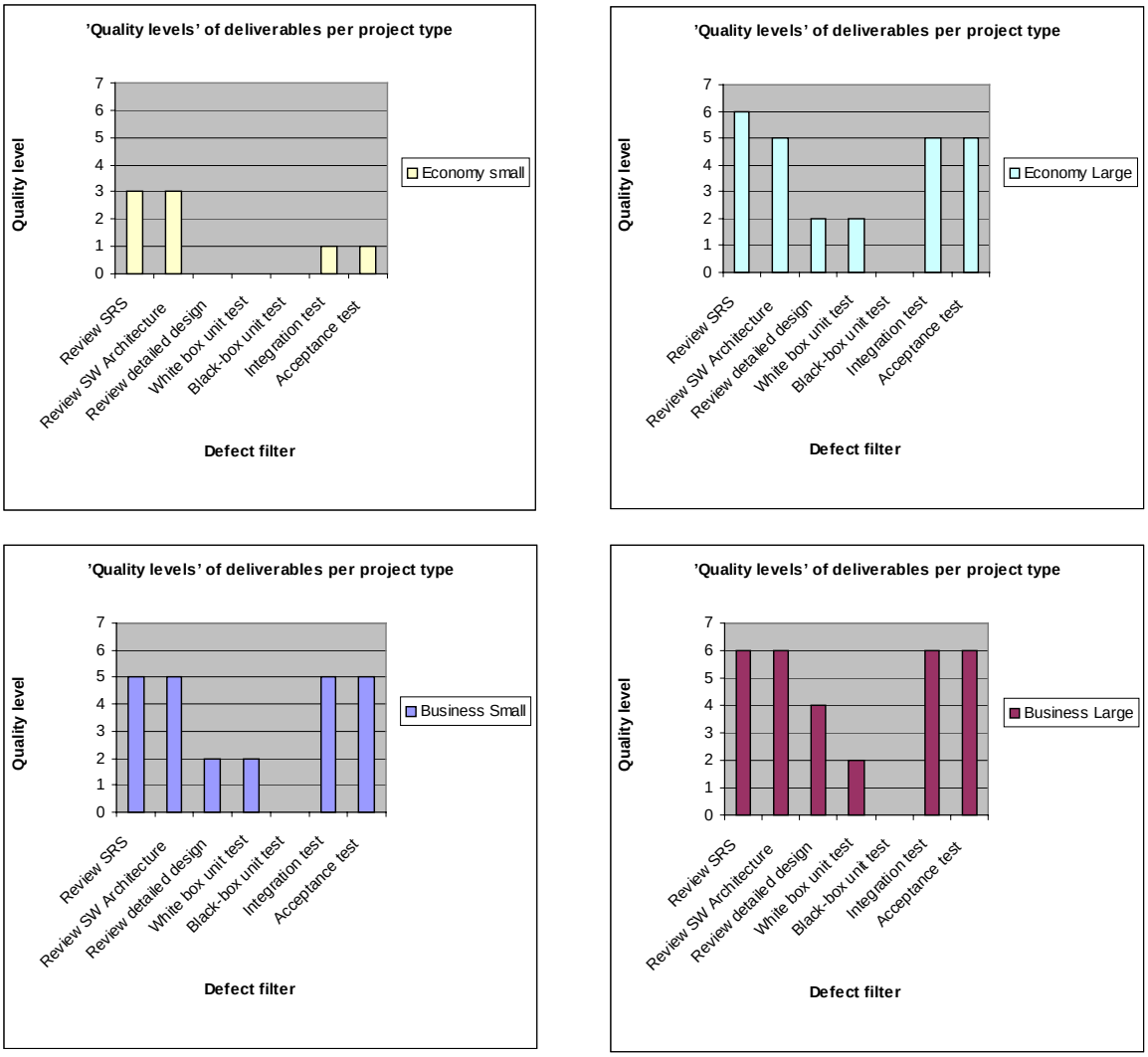

Figure 11: Quality level in relation to project tailoring. The graphs show the effort spent for the activities indicated.

Table 13 Project type dependent Quality requirements for documentation

Deliverable	Type / Size	Business			Economy		
		Small	Medium	Large	Small	Medium	Large
Commercial Requirement Specification		M	M	M	C	C	M
Functional Requirements Specification		M	M	M	C	C	M
Hardware Software Interface		M	M	M	C	C	M
Software Architectural Design		C	M	M	-	O	O
Software Detailed Design		C	M	M	-	-	O
Software Unit Test Specification		O	O	C	-	-	O
Software Unit Test Report		O	O	C	-	-	O
Software Integration Test Specification(in system test specification)		O	M	M	-	-	C
Software Integration Test Report		O	O	M	-	-	C
Software Acceptance Test Specification		M	O	M	-	O	C
Software Acceptance Test Report		M	O	M	-	O	C
User documentation		M	C	M	O	O	O

Legend: Quality requirements for documentation. M = mandatory, C = conditional, O = optional.

7.5 Product Quality characteristics.

The commercial requirements specification defines the quality criteria for the product.

This information is input for the product development. Omission of one may have a huge impact on the project activities.

Table 14 Product Quality characteristics

Level / Characteristic	1	2	3	4	5
1. Suitability	Software performs simple well-defined function	Software performs multiple operations in well-defined environment	Software performs multiple operations in adaptable environment for single user	Software performs multiple operations in adaptable environment for multiple users	Re-configurable software in adaptable environment for multiple users
2. Interoperability	Isolated Application	Interaction with two or three specific systems	Interaction with specified range of systems	Interaction with range of systems from different sources	Interaction with any system which uses or provides specified services
3. Maturity	No risk if s/w fails	Recoverable loss or damage to property or reputation in case of failure	Irrecoverable loss or damage to property or reputation in case of failure	Possible physical harm to person in case of failure	Possible death in case of failure
4. Fault tolerance	No tolerance to any fault	Robustness to user's errors	Data integrity in case of failures	Robustness to software faults	Robustness to hardware failures
5. Recoverability	No economic risk through failure	Negligible economic risk through failure	Serious economic risk in case of failure (company affected)	Large economic loss in case of failure (company endangered)	Financial disaster in case of failure (company may not survive)
6. Availability	Infrequent usage – on the spot request	Usage on working days – on the spot request	Continuous usage during working days	Continuous usage every day, 12 hours a day	Continuous usage every day, 24 hours a day
7. Understandability	Used by technicians	Used by experts of the application domain	Used by trained people	Used by anyone, limited by country	Used by anyone in the world
8. Learnability	Single dedicated, frequent user	Single user, randomly	Multiple users	Multiple users with peaks in usage	Multiple, geographically distributed users
9. Operability	Command driven	Menu driven	Subset of graphical user interface	GUI, with usage of pointing devices	Multimedia
10. Time behaviour	Non-interactive systems	Interactive systems	Fast response interactive systems	Real-time Strict real-time (within milliseconds)	Resource utilisation
11. Negligible	Significant data volume	Big data volume; communication lines	Huge data volume; dedicated communication lines – Multi-users with high peaks	Multi-processor. Geographically distributed with high peaks	Analysability
12. Correction deadline	Correction deadline is next release	Correction deadline one month	Correction deadline one week	Correction deadline immediate	Changeability
13. Evolution style by release.	once a year	twice a year	Three to six releases a year	Continuous maintenance	Adaptability
14. Product designed for unique hw / sw platform	Unique but adaptable hw/sw platform	Some similar hw/sw platforms	Many similar hw/sw platforms	Product designed for range of significantly distinct hw/sw platforms	
15. Installability	Installation performed by supplier	Installation performed by trained	Installation performed by qualified users	Installation performed by experienced users	Installation performed by naïve users

Level / Characteristic	1	2	3	4	5
		technicians			

8. REQUIREMENTS MANAGEMENT INFORMATION MODELS.

8.1 The Component environment

There are marked differences between component-bases systems and non-component-based systems. Especially when requirements are changed, the impact of the change may affect the systems in which a component is used. Changing constraints, e.g., the resource usage could imply that the resource constraints as specified for a system are violated. Other aspects are the release plan of the component: schedule, its contents (which services are provided etc.), on which platform(s) or application(s) the component is validated or will be validated. The platform includes the system's architecture and hardware (requires list). Figure [] shows the different types of components and their dependencies on other components.

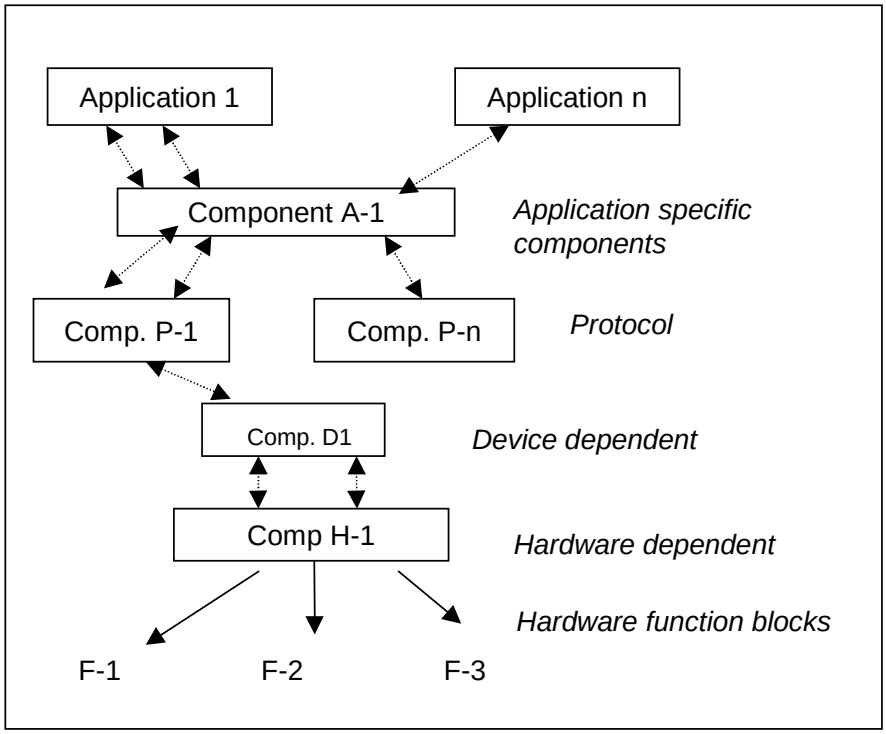

Figure 12 Components environment..

8.2 Document traceability structure.

During the product development the system level documents and the software level documents are created, possibly using different processes. The requirements management activity is done at every phase of the product development. The documents are owned / controlled by each process area. The traceability is done from [16] UR, FRS, SSD, to SRS etc, and source code up- and down -stream.

For each level, the input requirements are transformed to the level-dependent derived requirements.
Employees that work on a higher level specification fulfil the role of the external stakeholder for the next specification level.

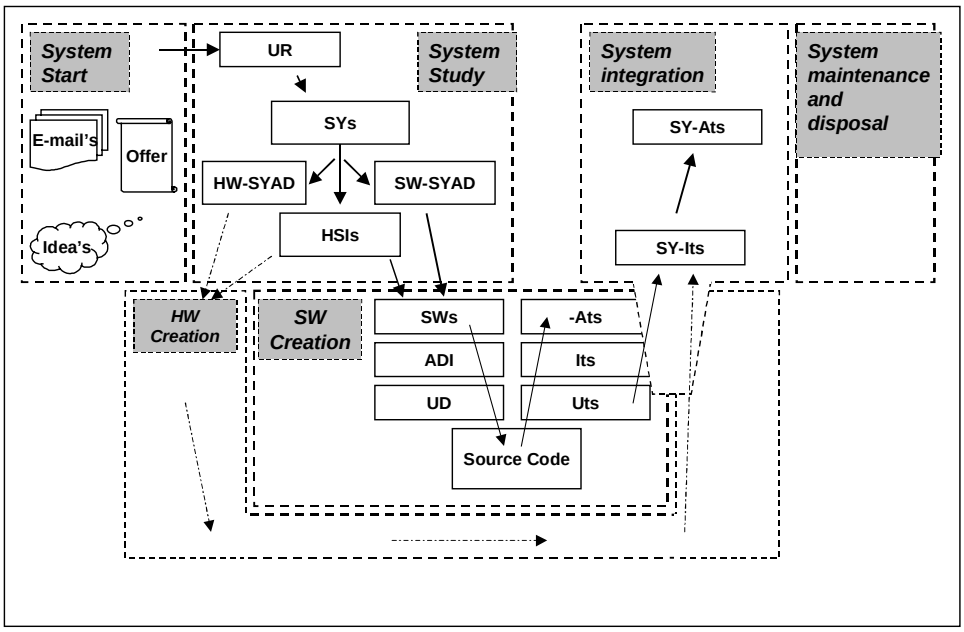

Figure 13 Document traceability structure.

The document information model is created to define relations between documents. The traceability between the documents and object types is documented as part of the requirements management plan.

8.3 The Requirements Management Information Models

The requirements management information models defines the information available and the traceability items and their relations.

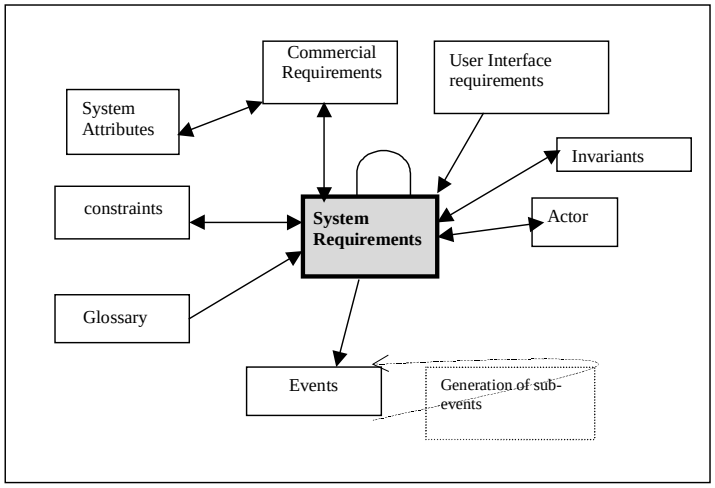

Figure 14 Traceability item overview

8.4 Component information model

8.4.1 General

High-level definition of a component

A ***component*** is a logically highly cohesive, lowly coupled, documented software module that can be used as a unit of development, reuse, composition and adaptation. It therefore is an exchangeable architectural element of a software system that acts as a part within a larger whole. It provides dedicated functionality that can be used in a specific application environment

When a component is reused for 100% in a different product or in the product line, then all related requirements, (requirements) constraints, quality status, development status, actual realised behaviour in terms of constraints (memory, performance) are reused and should be traceable in order to perform an adequate impact analysis.

The component information model shows the information of a component, which is relevant for the several stakeholders like architects, project managers, V&V engineers. This model shows the traceability items, which are to be traced to the component.

1. General product information (status, constraints, quality and product family)
2. For each specification or requirement: review reports, status, change requests and problem reports
3. Component-related information such as designs, architectures and reusability status.

A short description of these information items is given below.

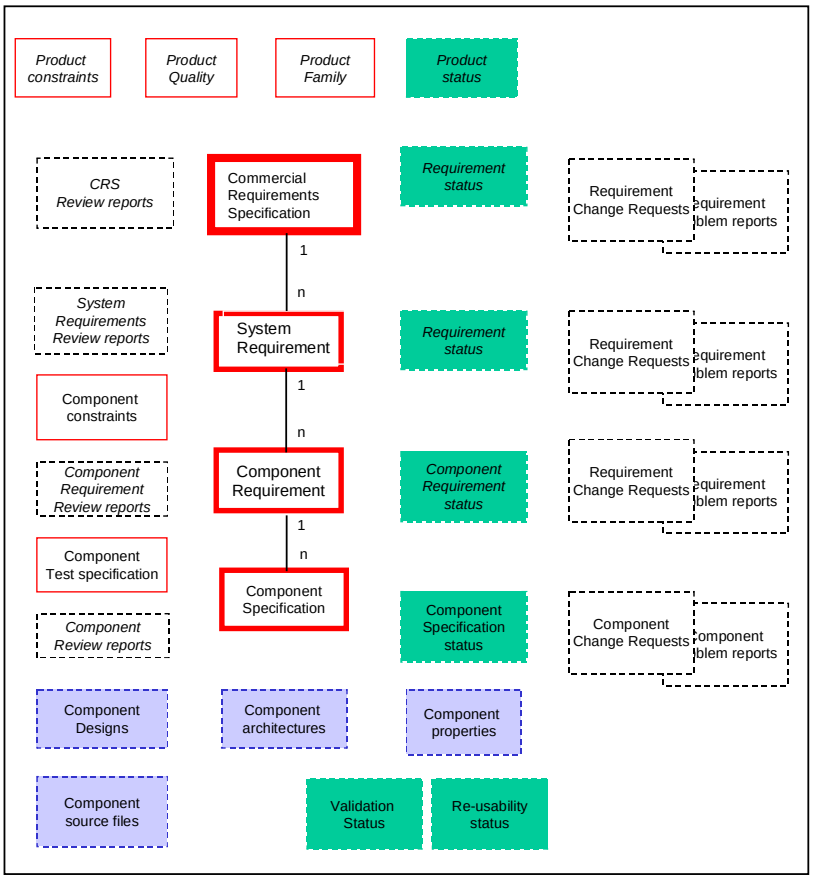

Figure 15 Component information model (global).

Review reports

The review reports describe the approval status of the document, the metrical data, like number of defects found and time spent on the review.

Change requests and problem reports

The list shows the history of a requirement and the status of the change proposals. The effects of the change requests, the effect on the lower level requirements or other specifications are traced.

Component Specification Status

General status:

1. The Configuration management version label and the reference to the version information like the release note.
2. The history of change by traces to problem reports.
3. The effort and costs spent.

Component Constraints

The constraints trace to the specified and actually realised constraint values.

Component Re-usability status

The re-usability status contains:

1. The list of applications for which the component has been validated.
2. The list of architectures the component has been validated.

Component validation status

Per platform and application test run, a status report shows:

1. Which types of tests have been performed.
2. Which test cases have been executed
3. The status of the functions tested.
4. Test coverage shows which provided services have been tested.

Component architectures

For the components architectural requirements refer to [5]Task 1.4 – Definition of Components and Notation for Components (D.1.4.2) *October 2000*

Component properties

1. The list of candidate architectures in which the component can be used.
2. Programming language used for implementation.
3. Constraints.
4. Actual performance in terms of resource usage.

9. TRACEABILITY OF REQUIREMENTS INFORMATION.

9.1 Introduction

9.1.1 General

Term ‘traceability’ covers the following aspects:

1. Traceability establishes links between the various elements of a product. It is insured by coherent requirements and imposes relations between documents.
2. Traceability is used to, at any time, remember and restore the history, the state, the location, the definition (requirements), the evolutions of a product and the functions which it carries out.
3. Traceability is required for the evaluation of the factors of maintainability, correctness, testability, and therefore contributes to the quality of a product.
4. Traceability is obtained by the strict and precise marking of the various elements of the product and by the systematic recording of their related events, i.e., it’s change history.

System requirements have to be traced in order to show that all the commercial requirements are met and that no unnecessary system requirements have been created. *Traceability* and change management are often identified as main issues within more general presentations on requirements management. Examples for definitions of traceability can be found in IEEE standard 830-1998 [6].
The relevant standards related to requirements management include requirements on the traceability of requirements and the execution of changes.
A common solution to guarantee the traceability is to file additional information within requirements attributes or independent elements. This additional information forms the set of parameters for the marketing- and technical managers, architects and engineers. This is sufficient to reconstruct the requirements development.

9.1.2 Up- and down stream

From the highest level of requirements specification to the lowest level of the product (source code) traceability is implemented such that automatic impact analysis is possible.

9.1.3 Interfaces and communication between specification levels

Requirements are exchanged between the specification levels by different interfaces. The information can be stored in a common database and if necessary with restricted access rights and traceability possibilities. Distributed locations often provide distributed databases. The data exchange is then facilitated by a similar structure of the information exchanged. The requirements can also be spread by electronic or paper documents which makes the use of requirements management tools more difficult if the updating of database contents corresponding to the documents treated requires a manual revision.

9.1.4 E-mail or other electronic information

E-mails or any other files related to the requirements specification are uniquely identified, filed and linked to the related requirements.

9.2 Techniques

Traceability can be established with different techniques. The technique used depends on the project size and tools available.
Predefined document structures dictate the contents of the project documentation and make traceability implicitly possible in this way.
Explicit cross-referencing is another possibility of establishing traceability.
Textual cross-references like “see section xy” are based on some form of information tagging, numbering or indexing.
Special cross-referencing schemes like traceability matrices are also an alternative.
Another technique is the enhancement of the project documentation by restructuring it in terms of an underlying network or graph.
Requirements management tools usually support explicit traceability between information in their databases. Traceability to external information (e.g. in external documents, in other tool databases) can be established via textual descriptions or explicit links in the tool’s database. In the latter case, an interface to the other tool is a prerequisite.

9.2.1 Manual Techniques

Reference to module traceability tables.

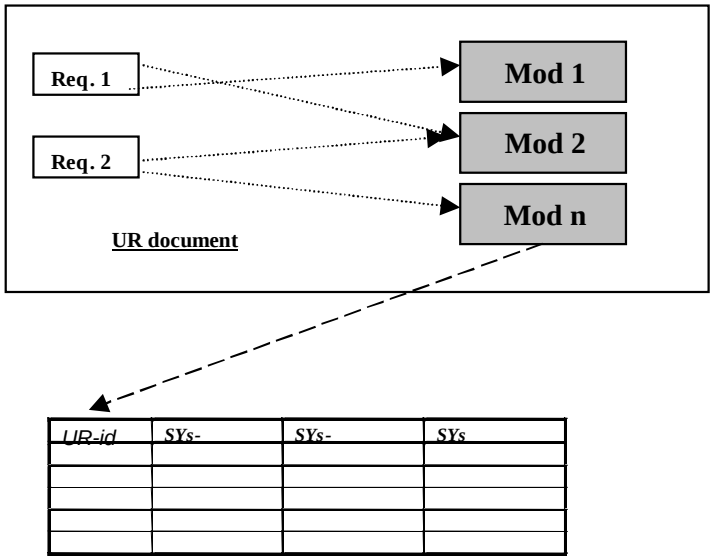

Figure 17 Traceability Technique : Manually between a UR and the Sys.

Per document, a cross-reference table is defined, in which the relation between a higher level requirement and the document defined specifications is noted.

9.2.2 Other techniques

Requirements storage structure

Requirements are either stored as:

1. individual database records. This increases the re-usability of the individual requirements. The generation of product-line products from a requirements database is then possible. Example are the requirement identifications of the UR to requirement identifications of a Sys.
2. part of a 'physical' document. Less flexibility is available for re-use and product—line definitions.

9.3 Impact analysis.

Impact analysis of the change of a requirement can be done by visually inspecting the related traceability items.

9.3.1 Constraints analysis.

The impact analysis of the constraints e.g. a memory constraint is to be supported by the analysis/design tools.

9.3.2 Value analysis.

The attribute values of a requirement can be analysed by defining special queries on the requirements management database.

10. CLASSIFICATION

Analysis of the elicitation information and classification of individual requirements obtain the requirements. The following classes are distinguished:

1. Functional requirements and data requirements.
2. Functional requirements.

Functional requirements are the fundamental subject matter of the system and are measured by concrete means like data values, decision-making logic and algorithms. Functional requirements refer to:

a) the behavior of the system
b) the functions which the system performs
c) the data exchanged with external systems with which it interacts once installed in the final environment.

10.1 Data requirements.

A specification of the essential subject , matter/business, objects/entities/classes that are germane to the system. This might take the form of a first-cut data model, an object model or a domain model. This model is used as input for the design of the components later on.

10.2 Non-functional requirements

Non-functional requirements are the behavioural properties that the specified functions must have, such as performance, usability, etc. Non-functional requirements can be assigned a specific measurement.

Table 15 Non functional requirement types

Look and Feel Requirements	**Examples/description**
The interface	The intention is to state requirements relating to the interface. *The product shall use the company colours*
The style of the product	The look and feel requirements specify the your client's vision of the product's appearance. *The product shall have a conservative and professional appearance.*

Usability Requirement	**Examples/description**
Ease of use.	The product's usability is derived from the abilities of the expected users of the product and the complexity of its functionality. *The product shall help the user to avoid making mistakes*
Ease of learning.	*The product shall be easy for an engineer to learn.*

Performance Requirements	Examples/description
Speed requirements	Specifies the amount of time available to complete specified tasks. Response time to events
Safety critical requirements	Quantification of perceived risk of possible damage to people, property and environment.
Precision requirements	Quantification of the desired accuracy of the results produced by the product.
Reliability and Availability requirements	This section quantifies the necessary reliability of the product. This is usually expressed as the allowable time between failures, or the total allowable failure rate. Refer to .Product Quality characteristics. On page 27. *The product shall be available for use 24 hours per day, 365 days per year*
Capacity requirements/	This section specifies the volumes that the product must be able to deal with and the numbers of data stored by the product.

Operability Requirements	Examples/description
Expected physical environment	Specifies the physical environment in which the product will operate. *The product shall be used in noisy conditions*
Expected technological environment	Specification of the hardware and other devices that make up the operating environment for the new system.
Partner applications	Description of other applications that the product must interface with

Maintainability and Portability Requirements	Examples/description
How easy must it be to maintain this product?	Specifies the maintenance aspects of the product e.g. if it is to be maintained by developers who are not the original developers. This has an effect on the way that the product is developed, and there may be additional requirements for documentation or training.
intended release cycle	Dates for the alpha, beta releases.
Portability requirements	Description of other platforms or environments to which the product must be ported.

Security Requirements	Examples/description
confidentiality	Specification of who has authorised access to the system and under what circumstances that access is granted.
Audit requirements	Specification of the required audit checks.

Cultural and Political Requirements	Examples/description
	This section contains requirements that are specific to the sociological and political factors that affect the acceptability of the product. If you are developing a product for foreign markets then these requirements are particularly relevant.

Legal equirements and Standards	Examples/description
Does the system fall under the jurisdiction of any law?	A statement specifying the legal requirements for this system.
Are there any standards with which we must comply?	A statement specifying applicable standards and referencing detailed standards descriptions. *The product shall comply with MilSpec standards."*

Verification requirements	Examples/description
	Defines the principles on how the system will be tested. *Verification is done by inspections/reviews or test, on component test or system level..*

Fit Criteria	Examples/description
	The *fit criterion* is an objective measure of the requirement's meaning; it is the criterion for evaluating whether or not a given solution fits the requirement. If a fit *criterion* cannot be adequately specified, then the requirement is ambiguous, or ill understood. If there is no fit *criterion*, then there is no way of knowing if a solution matches the requirement.

10.3 Product/project Constraints.

Table 16 Product / project constrains.

Mandated.	Examples/description
Solution constraints.	This specifies constraints on the way that the problem must be solved. *The product must use the Windows NT operating system.*
Implementation environment	This describes the technological and physical environment in which the product will be installed.
Partner applications	This describes applications that are not part of the product but with which the product will collaborate
Commercial off the shelf packages	This describes applications that must be used to implement some of the requirements for the product. Can ready-made components be used for this product? Is there something that we could copy?
Anticipated workplace environment	This describes any features of the workplace that could have an effect on the design of the product. *What problems could the new system cause in the current environment?* *Will the new development affect any of the installed system?*
Project milestones	Any known deadlines, that have an effect on product requirements.
Project Budget	
Product line requirements	

While other classifications of requirements are possible, developers should use this classification unless there are very good reasons to add classifications.

11. REQUIREMENTS IDENTIFICATION

Table 17 Requirement identification

Element	***Description***
<req-identifier> ::=	**[**<req. type prefix>[__<component name>]< reqnr>**]**
< req. type prefix > ::=	Example: *CRS* | *FRS* | *SAD* | *SRS* | *SATS* | *SITS* | *SUTS* | *SATR* | *SITR* | *SUTR* Note: When a tool is used, the **<** req. type prefix> can be configured for each requirement type or traceability item.
< component name > ::= <string>	Should be unique for a certain component of a system.
<req. nr> ::=	<integer>{.<integer>}* <integer> ::= {**1**, **2**, **3**, **4**, ...} So not zero! The dot-notation is only used when a requirement is split-up into sub-requirements. The numbers must be unique -- for a given prefix -- and persistent (don't change once created and don't re-use if deleted earlier). So, **[SRS3-F3]** and **[SRS3-N3]** is OK. *Note: When a tool is used the numbering depends on the tool*
Examples of valid references:	*[FRS_I2C.1.2], [SAD_DIVA3],*

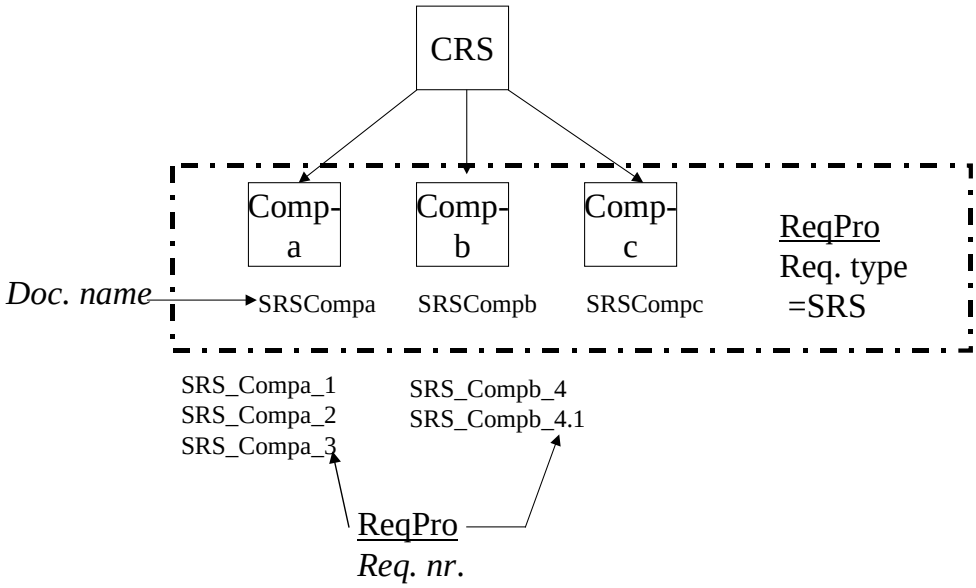

Figure 18 Example of a component requirement identification.

12. REQUIREMENT ATTRIBUTES

It is often necessary to capture additional information about individual requirements or groups of requirements. This additional information referred to as requirement attributes, further describes the requirement or imposes additional constraints on the development. . Existing attributes for requirements can also be taken from standards. The attributes of a requirement type, of a component and of the product used in a project are to be documented in the Requirements Management Plan. See: Project Management Plan on page: 18.

An attribute of a requirement provides additional information of that requirement.

1. the agreements made by the various requirements management stakeholders.
2. the statuses of a requirement.
3. the type of a requirement
4. various other things such as priority.
5.

The decision of administrating the attribute depends on the size of the project to be developed, on the level of detail of the specification of the requirement and on common sense. The attributes are in fact a checklist, which is used when the requirement is reviewed for completeness. Where the traceability shows e.g. the coverage of the UR requirements by the Sys requirements, the attribute values provide information on which the judgement of the reviewer is based. The attributes and their values can be used in automatically analysis of any aspect of the system and/or components.

12.1 General set of attributes

Each requirement has a fixed set of attributes.

Table 18 Attributes general for each requirement type

Name	*Description*	*Type*	*Values*
Absolute reference	A unique unchanging identification	Tool dependent	<doc-id> <number>
Stability	Is the requirement solid enough to start the next development phase?	Enum	Yes, No, not decided yet
Change status	When a change is made at the lower or higher level specification, the status becomes “obscure”, indicating that the impact inspection should be performed. After the impact has been reviewed, this status is cleared to consistent.	Enum	Consistent, Obscure, New
Requirement type	See Classification on page 38	Enum	1. Functional, 2. Usability Requirements 3. Performance Requirements 4. Operability Requirements 5. Maintainability and Portability Requirements 6. Security Requirements 7. Cultural and Political Requirements

Name	Description	Type	Values
			8. Legal Requirements and Standards 9. Product/project Constraints Mandate
Stakeholder	Who is the end-user/stakeholder of this requirement?	Enum	name1, name2, ...
Urgency	Implementation moment	Enum	<date-list>
Requirements catagory		Enum	Basic, performance, mandated, conditional, delight
Priority	Importance of requirement (relative to others)	Enum	Low, Mid, High
Variation	Is this a requirement, which is expected to change in the near future?	Enum	Low, Medium, High, not decided yet
Owner	Name of responsible person who is authorised to change the requirement	Enum	name1, name2, ...
Source	Name of author of the requirement	Enum	name1, name2, ...
Comments	Free text	text	

12.2 Customer Requirements

12.2.1 Business related Requirements attributes

Name	Description	Type	Values
Budget	Budget margin		
Urgency	Implementation moment	Enum	1999-2000, 2000-2001, 2001 and after
Accepted (customer)	Has the customer accepted the requirement?	Enum	Yes, No, not decided yet
Accepted (project)	Has the project team accepted the requirement?	Enum	Yes, No, not decided yet
Product quality characteristic	See: On page 48	Enum	
Disadvantage	Parameter for computing the priority		1 (= low) to 9 (=high)
Importance	Parameter for computing the priority		1 (= low) to 9 (=high)
Technical risk	Parameter for computing the priority		1 (= low) to 9 (=high)
Development costs	Parameter for computing the priority		1 (= low) to 9 (=high)
Priority			Priority = (importance + disadvantage) / (costs + risk)

12.2.2 Analysis of Customer Requirements

Name	Description	Type	Values
Priority	Importance of requirement (relative to others)	Enum	Low, Mid, High
Owner	Name of responsible person who is authorised to change the requirement	Enum	name1, name2, ...
Source	Name of author of the requirement	Enum	name1, name2, ...
Necessity	Defines the degree of necessity	Enum	Essential, Conditional, Optional
Comments	General remarks	Text	<...>
Questions	General questions regarding the requirement	Text	<...>
Is Requirement	Is the object a requirement or something else (header text, introductory text, …)	Enum	Yes, No, not decided yet
Effort			
Costs			
Reuse	Does this requirement imply that any objects will be reused?	Enum	Yes, No, not decided yet
Performance		text	
urgency	How soon is the requirement needed.		

12.2.3 V&V related attributes of a Customer Requirement

Name	Description	Type	Values
Acceptance Criteria	What is the nature of the test that would satisfy the user that the requirement is met	text	
Verifiable		Enum	Yes, No, not decided yet

12.3 System/Software Requirements Specification

Name	Description	Type	Values
Software Release	Implementation moment	Enum	Date1,date2, ...
Priority	Importance of requirement (relative to others)	Enum	Low, Mid, High
Source	Name of author of the requirement	String	<...>
Accepted (project)	Has the project team accepted the requirement?	Enum	Yes, No, Undefined
Questions	Questions to the author regarding the requirement	Text	<...>
Reuse	Does this requirement imply that any objects will be reused?	Enum	Yes, No, Undefined

12.3.1 Test attributes of a System/Software Requirements Specification.

Name	Description	Type	Values
Acceptance test stage	Has the customer ordered this test stage.	Enum	Yes, No, not decided yet
Integration test. stage	Has the customer ordered this test stage.	Enum	Yes, No, not decided yet
Unit test stage	Has the customer ordered this test stage.	Enum	Yes, No, not decided yet
Test Stage progress	Indicates the test stage done.	Enum	Acceptance test, Integration test., Unit test,
Review	Status of the review of this requirement	Enum	Untested, In-progress, Finished
static test	Status of the static test of this requirement	Enum	Untested, In-progress, Finished
dynamic test	Status of the dynamic test of this requirement	Enum	Untested, In-progress, Finished
white box test	Status of the white box test of this requirement	Enum	Untested, In-progress, Finished
Black box test	Status of the black box test of this requirement	Enum	Untested, In-progress, Finished

12.4 Test Specifications.

The test cases described are traceable to there related requirement(s).

12.4.1 All Test Specifications/Plans

Name	Description	Type	Values
Test status	Status of the test	Enum	Untested, In-progress, Finished
Comments	General remarks	Text	<...>
Accepted (customer)	Has the customer accepted the test specification?	Enum	Yes, No, not decided yet
Accepted (project)	Has the project team accepted the test specification?	Enum	Yes, No, not decided yet

12.4.2 All Test Results.

The test results are traceable to there related test cases.

Name	Description	Type	Values
Test result	Result of the test	Enum	Passed, Failed, Untested
Comments	General remarks	Text	<...>

APPENDICES.

APPENDIX 1.1 Requirement Guidelines

The following sections provide basic guidelines to be used when developing requirements. These requirement attributes are sufficiently objective such that a person with limited knowledge of the system's domain can usually determine compliance.

1. Requirements Must Be Concise and Singular (not Compound)

The requirement statement must include one and only one requirement. The requirement must state what must be done and only what must be done, stated simply and clearly. It must be easy to read and understand.
Compound requirements are not individually identifiable – a basic assumption of applying the remaining criteria. When a compound requirement is used, each requirement must be uniquely identified. For example, one way to specify the content of a label would be as follows:

Table 19 Example of a requirement identification (part of manual technique.)

SY-compa-970 SOURCE: CRS-070;	The prduct label shall contain the following information: a) manufacturer name b) serial number c) model number d) date of production e) country of manufacture

2. Requirements Must Be "Implementation Free"

A requirement must state only what is required without specifying how the requirement should be met. The requirement statement must not reflect a design or implementation nor describe an operation. A common problem is requirement statements that detail the implementation aspects while neglecting to state the actual function or non-functional attribute being required.

3. Requirements Must Be Complete (Stand-alone)

The requirement must be capable of standing alone when separated from other requirements. The requirement must be complete to provide sufficient information for testing without further amplification. Requirement statements are often pasted into Test Specs and other documents. Information in the requirement should allow the statement to be understood in this context.

4. Requirements Must Be Consistent

The stated requirement must be clearly derived and must not contradict requirements in the source documents.

5. Requirements Must Be Unambiguous

The requirement must have only one interpretation. This criterion must be considered within the purpose (definition) of the document and the intended audience.
It is highly recommended that "Standard Constructs" be used to provide structure to the natural language. For example, use the words "shall" or "must" to denote a binding requirement and avoid ambiguous words and phrases like "support", "adequate", "fault tolerant", "user friendly", and "as much as possible".

6. Requirements Must Be Verifiable

The requirement must be quantified in a manner that can be verified by objective analysis or test.
The requirement always has a fit criterion.
This *fit criterion* is an objective measure of the requirement's meaning; it is the criterion for evaluating whether or not a given solution fits the requirement. If a fit *criterion* cannot be adequately specified, then the requirement is ambiguous, or ill understood. If there is no fit *criterion*, then there is no way of knowing if a solution matches the requirement.

Requirement Specifications Document Review Checklist

Table 20 Requirements review checklist

Check Item		Status (Y/N)	Remarks
1. Is the source document approved?			
2. Are all requirements uniquely identified?			
3. Are all the requirements of the correct type			
4. Are all applicable requirements of the source document covered in this document?			
5. Does the document cover requirement for all possible scenarios?			
6. Is timing or other constraint specified?			
7. Are all requirements identified in this document traceable to the documents as specified in the information model?			
8. Are all the requirements consistent amongst themselves?			
9. Does the specification include non-requirements (i.e. What the software should not do)?			
10. Is the entire requirements consistent vis-à-vis existing system behaviour?			
11. Are all requirements correct with respect to the applicable standards / source documents?			
12. Do all requirements have unique interpretation?			
13. Are all requirement statements free of words – "AND", "OR" & "NOR"?			
14. Are all requirements granular?			
15. Per function: *Functional requirements should define the fundamental actions that must take place in the software in accepting and processing the inputs and in processing and generating the outputs.*	*1. validity checks on the inputs* *2. sequence of operations* *3. responses to abnormal situations, including* • *overflow* • *communication facilities* • *error handling and recovery* *4. effect of parameters* *5. relationships of outputs to inputs, including* • *input/output sequences* • *formulas for input to output conversion*		
16. Are all requirements testable?			
17. Are all requirements feasible?			
18. Is the document in line with the required standards?			

14. STANDARDS.

The requirements management is based on the relevant standards:

1.	CMM
2.	DOD2167A
3.	ARP 4754,
4.	DO-178B,
5.	PSS-05 or the Development Standard for IT-Systems of the Federal Republic of Germany (V-Model).

CMM Model Requirements

The documentation of traceability between assigned requirements and software requirements, design, code and tests data, is managed and checked.
The term documentation does not necessarily mean e.g. a Word document, but every registered item.

DOD2167A Requirements

The standard asks to trace the SRS and IRS software requirements with:

- system requirements assigned to the software (SSDD),
- tests (STD),
- the CSC and CSU, without any description of the way to achieve it.

Requirements of the DO178

The following text is an extract from the standard.

The directives of traceability are the following ones. :

- The traceability between the specifications of the system and those of the software must be provided to verify the complete system specifications and to give a visibility of derived requirements.
- The traceability between the low level requirements and the high level requirements has to be provided to verify derived requirements and choices on the architecture made during the software design, and to allow the check of all the high level requirements.
- The traceability between source code and the low level requirements has to be provided to check the lack of not documented source code and the check of all the low level requirements.

APPENDIX 1.1.1.1 Definitions of traceability

IEEE standard 610.12-1990 :

traceability.
(1) The degree to which a relationship can be established between two or more products of the development process, especially products having a predecessor-successor or master-subordinate relationship to one another; for example, the degree to which the requirements and design of a given software component match. *See also:* **consistency**.
(2) The degree to which each element in a software development product establishes its reason for existing; for example, the degree to which each element in a bubble chart references the requirement that it satisfies.

IEEE Std 830-1998
IEEE Recommended Practice for Software Requirements Specifications

4.3.8 Traceable
An SRS is traceable if the origin of each of its requirements is clear and if it facilitates the referencing of each requirement in future development or enhancement documentation. The following two types of trace-ability are recommended:

3. *Backward traceability (i.e., to previous stages of development).* This depends upon each requirement explicitly referencing its source in earlier documents.
4. *Forward traceability (i.e., to all documents spawned by the SRS).* This depends upon each requirement in the SRS having a unique name or reference number.

The forward traceability of the SRS is especially important when the software product enters the operation and maintenance phase. As code and design documents are modified, it is essential to be able to ascertain the complete set of requirements that may be affected by those modifications.

IEEE standard 830-1998:

A software requirements specification is **traceable** if (i) the source of each of its requirements is clear and if (ii) it facilitates the referencing of each requirement in future development or enhancement documentation.

Programme Management and System Engineering Capability Maturity Model

PA 02 - Derive and Allocate Requirements

The major functions or services of the product or system being developed are identified. The critical requirements are identified and analysed (in terms of technical performance, costs, schedule and risks). The external constraints which are not explicit in the customer requirements are identified and analysed (including political environment, partnerships and interfaces with other programmes). All requirements are allocated to functions, objects and groups of individuals to enable solution assessment. The requirements are traced and their evolutions are managed. Validation checks are available for each function or requirement.

PA 09 - Manage Configurations

The method for managing configurations is defined (perhaps in the Configuration Management Plan). The configuration items are identified and configuration baselines (functional, allocated, product, and programme baselines) and associated documents are managed. The changes to established configurations are controlled (engineering change management and approval, waivers etc). The configuration status is established and communicated to all affected groups.