T U M

I N S T I T U T F Ü R I N F O R M A T I K

Requirements Engineering Reference Model (REM)

Eva Geisberger, Manfred Broy
Technische Universitaet Muenchen
Brian Berenbach, Juergen Kazmeier,
Daniel Paulish, Arnold Rudorfer
Siemens Corporate Research, Princeton

TUM-I0618
November 06

T E C H N I S C H E U N I V E R S I T Ä T M Ü N C H E N

TUM-INFO-11-I0618-00/1.-FI

Druck: Institut für Informatik der
Technischen Universität München

Requirements Engineering Reference Model (REM)

Main author:

Eva Geisberger

Software & Systems Engineering
Technische Universität München (TUM)

Contributing authors:

Manfred Broy (TUM)

Brian Berenbach
Siemens Corporate Research
Princeton (SCR)

Jürgen Kazmeier (SCR)

Daniel Paulish (SCR)

Arnold Rudorfer (SCR)

Final Version

OCT 31 - 2006

Preliminary Remark

The following technical report of the Technische Universität München (TUM) describes the Requirements Engineering Reference Model (REM). REM is the result of long-term research in the fields of systems and software engineering, model-based requirements engineering and process definition of the chair of Software and Systems Engineering at the TUM.

This report is the conclusion of the ongoing cooperation with Siemens Corporate Research in Princeton (SCR). It is almost identically to the corresponding Siemens REM-report. Simply Siemens-specific notes are taken out.

Contents

Executive Summary

Requirements engineering (RE) is a key technology for the successful development of software-intensive systems and products. Thus, RE is a critical capability for such enterprises businesses. There are many crucial ingredients and success factors for RE. Among others, a framework is needed which can be used to develop an RE methodology as part of the initial steps towards achieving an efficient RE organization and process.

This document describes a RE framework called the Requirements Engineering Reference Model (REM). REM provides:

- a core set of RE artifacts and their dependencies. Artifacts are the consolidated work results of the various RE activities in product or product-line development;
- a role model for the responsibilities in RE;
- a tailoring approach that helps guide software projects in setting up their RE specific processes and tools.

REM can be used for measuring, comparing, evaluating and improving current RE practices, methods, and tools. REM supports the development of software-intensive systems (from definition to maintenance), including upstream activities to explore system concepts. REM will enable enterprises to base their RE process improvement efforts on proven best practices, tools, and methods.

REM will include a comprehensive set of methods and tools to support its artifacts and activities. Enhancing REM in these directions is necessary for a comprehensive RE approach. Retrospective analysis of projects will lead to a collection of RE best practices.

The chair of Software und Systems Engineering at the Technische Universität München is partnering with Siemens Corporate Research Princeton , to develop REM. The REM approach is aligned and complimentary with CMMI process improvement activities, also with the product lifecycle management (PLM) activities at Siemens.

Chapter 1 Introduction

Defining the most appropriate system requirements is a key marketplace success factor for enterprises like Siemens or other system development organizations. Due to intense competition, time-to-market has become critical for organizations to achieve profitability, especially when they must make large investments for system development. The earliest phases -- product conception and definition -- have the highest impact on those enterprises bottom line. Therefore, RE excellence is a must capability for their businesses to be implemented in the medium- to the long-term.

RE addresses all aspects of capturing, analyzing, negotiating, deciding, structuring, prioritizing, and validating features and requirements for products. Currently, RE is more an art than an engineering discipline. It is a young, interdisciplinary activity involving many stakeholders within an organization.

There is currently no roadmap available for achieving RE excellence. There is no comprehensive well accepted RE approach addressing practical, methodological, and conceptual needs. Tool support is implemented inharmoniously with the processes the tools must support. Traceability of requirements is a huge challenge for systems development. A framework is needed, within which to implement the path to RE excellence.

One of the leading research institutions, the Technische Universität München (TUM), is partnering with Siemens Corporate Research Princeton (SCR) to develop a RE Reference Approach called the Requirements Engineering Reference Model (REM). This report is targeted towards researcher and practitioner in the field of systems and software engineering.

1.1 Core Content of REM

The purposes of REM are: (1) to define a reference model for RE that provides the core set of RE artifacts (work products) and their interdependencies, and (2) to guide the establishment and maintenance of product- and project-specific RE processes.

The central purposes of RE are to:

- Analyze marketing information and user needs to derive the functional and non-functional requirements governing functional system's design;
- Understand the effect of these requirements on the business that creates the product;
- Consolidate these requirements into a consistent and complete requirements and systems specification.

RE thus has a central role in product definition and development. The RE artifacts support product design decisions and project management throughout the entire product lifecycle. The quality and appropriateness of these artifacts is a key factor for successful system development. Developing a consistent and comprehensive specification of the “desired” system is the overall objective of RE.

Requirements engineering of complex systems involves several challenges:

- Achieving measurable quality requirements, such as appropriate functionality, safety, performance, security and usability.
- Appropriate integration of the system under consideration into existing (or developed in parallel) distributed systems with different application domains, disciplines and various control hierarchies.
- *Product lines* require, additionally:
    - The management of related sets of requirements for the products in a product portfolio;
    - Long-term evolution strategies for the product family;
    - Maintaining the requirements base over long periods of time, and;
    - Managing non-functional requirements unique to product lines, such as variability and standardization.
- Cost and return-of-investment (ROI) estimation techniques must be developed that predict the consequences of project-specific system design decisions to the overall development costs of a product or product-line and their long-term ROI.
- Global, distributed development and reduced time-to-market multiply communication problems in requirements engineering.

REM supports the engineering activities by providing:

- An *RE artifact model* as a measurable reference model for interdisciplinary communication and specification work;
- A *tailoring concept* that specializes the RE artifact model to specific project needs, and;
- *RE artifact-centered process guidance* that defines a set of *completion levels* of the RE artifact model. The specified completion levels form a baseline for measuring progress and quality control.

1.1.1 Reference Model-based Requirements Engineering

The core concepts of REM are:

- The common requirements engineering artifact model. This is a model of the requirement specifications and their dependencies. The artifact model serves as a basis for requirements negotiation (between customers, users and service providers, in the broadest sense). It guides requirements development and the measurement of project progress and quality control. It is described in Chapter 2.

- A model of an iterative requirements elicitation and consolidation process that is used to analyze and complete requirements and systems specifications. Measurement of progress and quality in requirements development is based on the predefined completion levels of the RE artifact model. The completion levels define decision gates such as milestones and quality gates in the development process. Chapter 3 describes this artifact-oriented process support.

1.1.2 Tailoring to Specific Process Models

Tailoring is the means to have on one side a standard approach to RE and to avoid at the same time the "one-size-fits-all" pitfall. Tailoring in REM is done to address the needs of specific application domains and project characteristics.

The REM provides an artifact-oriented tailoring concept, to define specific RE work products, document structures and process definitions for the needs of specific projects and domains. This includes the assignment of specific methods and description techniques to support communication and assure a proper level of formality. Tailoring is described in detail in Chapter 3.

In REM, tailoring is done by:

- **Trimming the artifact model** (cut and tune): REM provides a comprehensive set of RE artifacts. In individual projects, not all of these artifacts are necessary. During tailoring, it may be decided that certain artifacts are not needed. These are left out of the tailored RE process model.

  As part of trimming the REM artifact model, the RE artifact structure may also be reconfigured and repackaged. Certain artifacts may be combined into one document or split into several documents.

- **Selecting methods for developing and representing RE artifacts**: REM does not dictate the individual methods and techniques for developing, structuring, and representing the individual RE artifacts. Rather, REM provides lists of recommendations for these activities. These recommendation lists are not exhaustive, and they are expected to be enhanced in the future.

- **Deciding on the process**: REM does not dictate the sequence in which the RE artifacts are produced. This sequence is determined by the specified development process. REM recommends an iterative approach, but does not constrain it to this choice. The process determines not only the order the artifacts are developed, but also in what detail they are elaborated, and how the quality assurance process is executed when defining quality gates.

1.1.3 Roles, Responsibilities and Organizational Impact

The *REM role concept* defines the types of people involved in the RE activities. It is used to assign responsibilities within a development project. The role concept is defined within a clear organizational communication and decision structure.

During any RE process, decisions must be made and the artifacts generated by RE be reviewed. REM supports this decision process by a tailorable RE artifact model for making such decisions. The basis for review and product decisions within an individual project are specification documents tailored from the RE artifact model and other predefined project management document templates.

REM defines a stakeholder-oriented role concept that supports the necessary communication and decision making in RE by shared commitment for developing the RE artifacts with clear accountability (see Figure 1).

An assignment of roles to the RE artifacts is described in Chapter 2. For every REM artifact, the *responsible* and *contributing* roles are assigned. *Responsible* means "ensuring that the artifact gets completed", and *contributing* means that the role "has a stake in the specification, but isn't responsible for completing it".

Through tailoring individual project team members are assigned to the roles. A person can be assigned to more than one role, and a role can be filled by more than one team member. One team member is always identified as the role leader or document owner.

Role (Cluster)	Objective	Functional Area	RE Artifacts
Product Management (ProdM)	Delivery of cost-effective products and solutions that meet customer needs	Planning and managing the entire life cycle of a product, including identifying customer needs, system vision and scope	Business objectives, customer/user requirements, system vision, conditions and scope, product portfolio, return of investment (ROI), risks, system success factors
Requirements Engineering (RE)	Qualified and comprehensible/reusable product decisions	Refinement and analysis of business objectives, reasonable & consolidated modeling of customer/user and business processes (functional, domain, quality goals, constraints)	Analysis models of customer and business needs (functional, domain, quality goals, constraints), user interface and system specification, acceptance conditions
Systems Architecture (SA)	High-quality and cost effective system design that meets business requirements	Specifying system architecture according to quality & business requirements, defining the system structure decomposing the system into functional interface specifications	Comprehensible functional system specification, system integration and interfaces specification, release planning, system test criteria
Project Management (ProjM)	Delivering the product solution within project constraints	Planning & managing the product development, process definition, measurement and control	System specification, design constraints, risk analysis, process requirements and constraints
Development (Dev)	Build to specifications	Implementation of product solution, including (hardware) design, coding, integration, testing	System/interface specification, design constraints, integration plan, system test criteria
Quality Assurance (QA)	Ensure verified product quality	Review and measurement of all specifications according to domain-specific quality standards	Measurable specifications, system integration and (acceptance) test specification
Release Management (RelM)	Incremental release of product features	Release planning and execution according to market strategy, system structure, development sequence and integration	Release strategy, system specifications, release planning, corresponding system interface, integration and test specification

Figure 1: Overview of the REM Role Concept

1.2 Scope and Objectives of REM

REM supports the development of complex systems including early phase activities to explore system concepts. REM can be used for embedded systems where software is embedded in hardware devices and for enterprise software systems that support organizational or technical processes.

REM is tailored to organizations and projects according to their specific domain, technology and business needs. REM supports a wide range of process models for system development ranging from agile to conventional.

REM can be applied in a lightweight manner producing a limited number of informal or text-based documents but also in a heavyweight manner using formal methods and modeling techniques (depending on the business needs). REM is complemented by a number of existing and emerging methods for producing the RE artifacts.

Objectives of REM

- **Standard frameworks to establish scalable and effective RE processes, methods, and tools.** REM is a yardstick for measuring, comparing, and evaluating RE practices. Without such a yardstick, it is difficult for requirements engineers to accurately evaluate the strengths or weaknesses of RE practices in comparison with other RE practices.
- **Practically verified, explicitly defined basis for establishing, integrating, and tailoring RE methods.** Without such a basis, it is difficult for requirements engineers to understand how RE methods can complement each other and where they should or should not be applied to address specific RE issues.
- **An explicitly defined, integrated software and system development process model that defines the roles of RE within the overall development process and organization.** Without such a model, it is difficult for stakeholders to understand how RE fits into the overall product development life cycle and how software and system development organizations and projects are structured to effectively facilitate the use of RE methods.

1.3 Overall Document Structure

Chapter 2 introduces the RE artifact model – a reference model of RE work. The RE artifact model is divided into three parts: Business Needs, Requirements Specification and System Specification.

Chapter 3 describes the tailoring concepts of REM. The principles of an artifact-centered process support for achieving measurable quality control in development projects are discussed.

Chapter 4 outlines the methodological base of REM by showing the underlying structuring and modeling principles of the RE artifact model.

Finally, *Chapter 5* summarizes the REM approach and its benefits and gives an outlook to its further application.

The history and background of REM are summarized in *Appendix A*.

The underlying philosophy of RE as an iterative process of problem solving and its corresponding system modeling concepts are summarized in *Appendix B*.

Appendix C shows the TUM prototypical instance of REM using the data model of the integrated RE and system specification tool AutoRAID/AutoFocus. It is based on a mathematically founded system modeling concept and in addition allows formal techniques for assuring the quality of the developed specifications.

Chapter 2 RE Artifact Model

RE is an interdisciplinary and iterative engineering set of tasks, during which requirements are stepwise developed, defined and specified. This regards all kinds of requirements like describing business objectives, customer requirements or developed system solution concepts. The various specification documents are used to support the product design and management decisions at milestones throughout the entire product development life-cycle. Thus, the level of quality and appropriateness of these specifications relating to business goals and customer needs is a key factor for successful product development.

REM structures and guides the RE activities in terms of a common reference model of RE specification work products – the RE Artifact Model (see Figure 2). This chapter gives an overview of the REM RE Artifact Model and its structuring into major groups of artifacts. For each artifact its content and purpose in RE is described. Examples or references to methods, descriptions, and templates for working out the artifact, and to help guide instantiating the RE Artifact Model are provided.

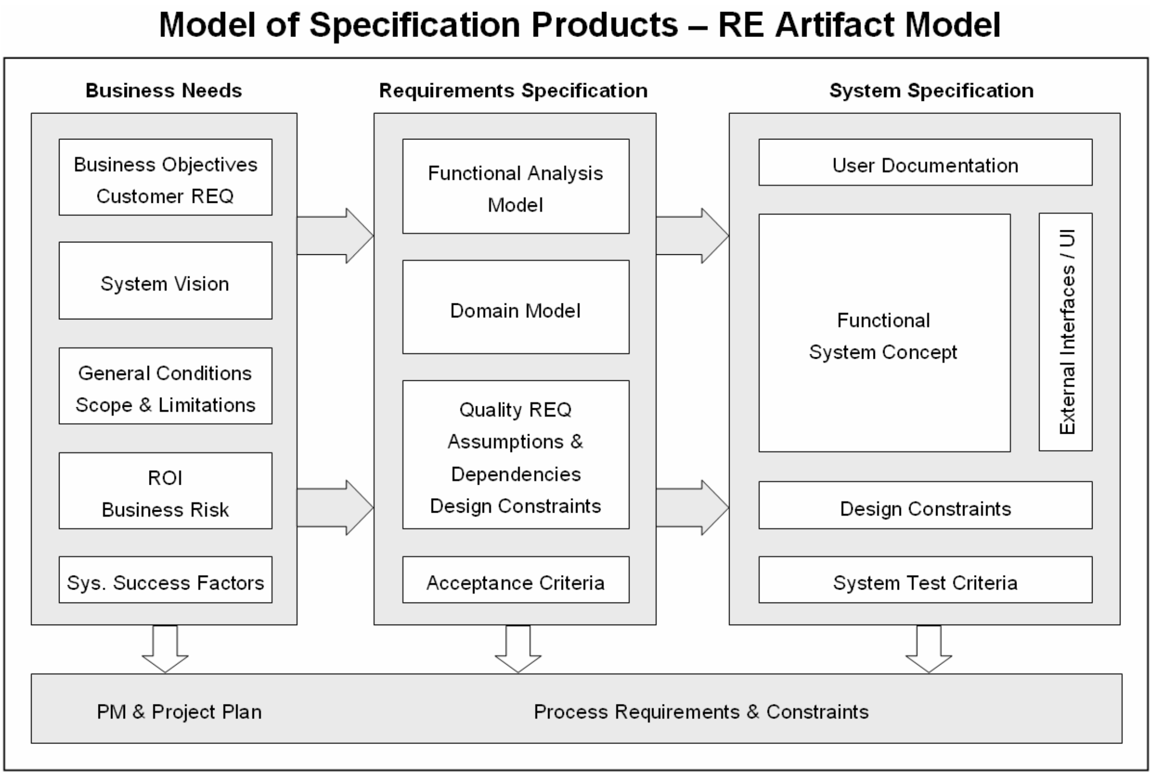


Figure 2: Overview of the Requirements Engineering Artifact Model

2.1 Requirements Engineering Artifact Model - Overview

The RE Artifact Model is structured into the groups: *Business Needs, Requirements Specification, and System Specification*. Figure 2 shows an overview of the RE Artifact Model, its groups, with artifacts and content items[1].

These three groups represent different abstraction-levels of requirements specifications of a system. The RE Artifact Model outlines the corresponding core deliverables.

The Artifact Model is structured according the following principles of integrated requirements analysis and system design:

- A goal- and user-oriented analysis and refinement of requirements and functional design concepts
- Analysis and refinements of „high-level", functional and non-functional requirements by a basic concept of how to describe systems and their behavior with functional system views
- Improving the quality of requirements based on the defined dependencies between different classes of artifacts in the RE Artifact Model. These dependencies form measurable consistency constraints of requirements, and can be used to verify and validate requirements and system solution concepts.

The RE Artifact Model is built according to these principles and its single artifact descriptions provide recommendations for methods and description techniques to communicate and derive their necessary content.

A comprehensive explanation of this underlying analysis structure of the RE Artifact Model and the supporting requirements analysis and management techniques for verifying and validating requirements specifications is outlined in Chapter 4.

2.1.1 Groups

The RE artifact model consists of three main groups of RE specification results:

The *Business Needs* specify customer and strategic requirements, including product and business goals of the system development. It consists of the following artifacts:

- *Business Objectives* and *Customer Requirements* – product market positioning and customer requirements
- *System Vision* – A list of main features and assumption/dependencies of the planned product or product-line
- *General Conditions* and *Scope & Limitations* – "High-level" non-functional requirements and the delimited scope of the application domain or product-line

---

[1] The content items of each RE artifact are shown in the overview-figures of the artifact groups: Figure 4, 5 and 6.

- *ROI* and *Business Risk* – Cost/benefit, expected sales revenue, development & launch costs, and risk analysis

- *System Success Factors* – how will the system be judged to be successful.

The *Requirements Specification* contains the product functional and non-functional requirements. They are analyzed and modeled from the customer and user perspective and derived (and justified by) from the *Business Needs*. The included artifacts are:

- *Functional Analysis Models* – Analysis and description models of the business and application processes/scenarios. They serve as a means to communicate the customer/user needs, derive the necessary system services, features and quality attributes, and they are basis for verifying the developed *System Specification*.

- *Domain Model* – Structured specification of the application domain and its characteristics, together with an operational environment model of the required system.

- *Non-functional Requirements Model* - Quality requirements, assumptions, dependencies, and design constraints.

- *Acceptance Criteria* – Specification of the acceptance criteria and testing of the deliverable system.

Many of these *Requirements Specification* artifacts are described in terms of models. Their major role in RE is to refine the “high-level” stated *Business Needs*, structure them, define measurable requirements, and map them to detailed system requirements of the corresponding product solution concept. They serve as a decision base for prioritizing and defining the product requirements and system design concepts.

The *System Specification* contains a detailed definition of the functional system concept; the required behavior of the considered system and its integration into the overall system and environment. It defines constraints to the detailed design and realization of the system (software, hardware – electrical, mechanical). The System Specification artifacts include:

- *User Interface Specification/User Documentation* – Description of how the user will use the system

- *Functional System Concept* – Detailed functional system requirements, which specify the required services, interaction, behavior, data and use constraints of the system. Together with the *External Interfaces Specification* it specifies the integration into the overall system or domain.

- *External Interface Specification* – Interface specification of relevant systems/components of the domain; interface definitions of used software and hardware components.

- *Design Constraints* – Limitations to the further detailed design and realization of the specified system concept in the single engineering disciplines.

- *System Test Criteria* – Acceptance criteria and test cases for system integration and validation.

The document structure for a specific product development project is subject to tailoring and the organization’s process definition.

These three groups of RE work products represent prevalent topics of the major phases in system definition, not documents. Specification documents as basis for milestone decisions can be cut across this RE Artifact Model. The required document structure of a development project is subject of tailoring and domain-specific process definition.

2.1.2 Reference Description of RE Artifacts

The REM RE artifacts are described by the structure shown in Figure 3.

Artifact group		
Artifact X		Mandatory
Responsible:	*Contributing*:	
Description:		
Content item		Mandatory Recommended Optional
Content item		
Content item		
Purpose:		
References:		
Artifact Y		
Responsible:	*Contributing:*	

Figure 3: Structure of RE artifacts description

The definitions of the fields in Figure 3 are given below.

Responsible/Contributing identifies the roles for producing the artifact. The REM role model is summarized in chapter 1.

Description gives a general content description of the artifact.

Content items describe the artifacts by their specific content.

Mandatory/Recommended/Optional assigns importance attributes to the artifacts as a basis for tailoring (see chapter 3 for tailoring).

Purpose describes the purpose of the artifact in RE and how the content is related to other artifacts in the Artifact Model (see Chapter 4 for the underlying analysis structure of the Artifact Model).

Reference gives links to potential methods, description techniques or example specification templates.

2.2 Business Needs Artifacts

Figure 4 gives an overview of the Business Needs group of artifacts. These artifacts describe the business and system goals of the development in terms of business opportunities, objectives, related customer requirements, and by a set of main features, assumptions, and dependencies of the product to be developed. The general conditions, scope and limitations of the product to be developed are summarized. The system success factors and potential risks are stated. The product goals and limitations are the result of elaborated marketing, portfolio and customer negotiations. Together with returning ROI and risk analysis, they set the stage for subsequent development work and form the rational for product design decisions.

Ongoing ROI and risk calculation is crucial for project success

At the beginning of a project, the features, assumptions and dependencies as well as general conditions are vague and incompletely defined. These requirements need to be clarified, refined and properly specified by the functional system modeling artifacts of REM. An ROI & risk calculation is a central artifact within the REM approach, and its ongoing revision based on the results of requirements clarification is a major tool to deal with the uncertainties and vague requirements in system definition.

The specific Business Needs artifacts are described below. Figure 4 provides an overview.

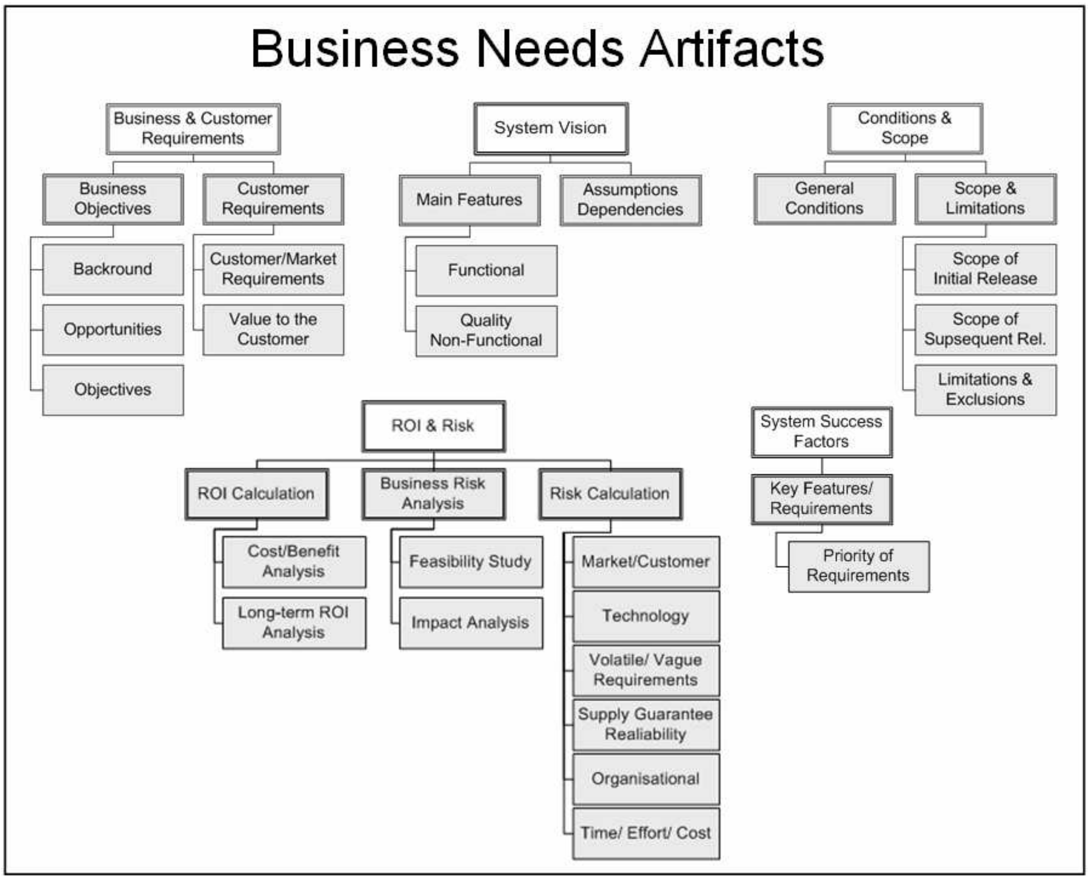


Figure 4: Overview of the REM Business Needs Artifacts

Business Needs Artifacts

Business and Customer Requirements		Mandatory
Responsible: ProdM	*Contributing:* RE, SA	
Description: The Business and Customer Requirements identify the primary benefits that the new system will provide to the customer and to the organization that is developing the system.		
Business Objectives		Mandatory
Summarize the important business benefits the system will provide, preferable in a way that is quantitative and measurable. The background and business opportunities of the future system are described. This includes a description of business problems that are being solved, and a comparative evaluation of existing systems and potential solutions. The rational for the system development is described, and how the system aligns with market trends or corporate strategic decisions is defined.		
Customer Requirements		Mandatory
Summarize the needs of typical customers or users. Customer needs are defined at		

a high-level for any known critical conditions, interface, or quality requirements. They provide examples how customer will use the system and identify the components (hardware and software) of the environment in which the system will operate. Explicitly define the value the customer/user will receive from the future system and how it will lead to improve customer satisfaction.
Purpose: Business and customer requirements serve as entry points to context analysis and the specification of the required features and characteristics of the *System Vision* and the definition of the general *Conditions & Scope* of the development. By identifying the business objectives, the situation and the critical conditions, collect business risks associated with the developing (or not developing) this system systematically as input to risk and cost/ benefit analysis (*ROI & Risk*).
References: [Wie99] gives an overview of business requirements and provides a list of possible customer values.

Business Needs Artifacts (cont.)

<table>
<tr><td colspan="2">System Vision</td><td>Mandatory</td></tr>
<tr><td>Responsible: ProdM</td><td colspan="2">Contributing: RE, SA</td></tr>
<tr><td colspan="3">Description: Establishes a long-term vision for the system that addresses the business objectives. It provides the context for decision making throughout development.</td></tr>
<tr><td colspan="2">Vision Statement</td><td>Mandatory</td></tr>
<tr><td colspan="3">Summarize the long-term intent and purpose of the new system. This artifact is an overview description of the system development goals and functions, and it describes existing or anticipated customer markets, system families, domain architectures, strategic directions, and resource limitations. It forms an anchorage for deriving Features, Assumptions and Dependencies.</td></tr>
<tr><td colspan="2">Main Features</td><td>Mandatory</td></tr>
<tr><td colspan="3">This artifact lists the major features or user capabilities the new system will provide. These can be functional features, quality goals, or further non-functional requirements such as legal requirements or compatibility to specific domain standards.</td></tr>
<tr><td colspan="2">Assumption and Dependencies</td><td>Mandatory</td></tr>
<tr><td colspan="3">List any assumptions that were made when designing and planning the system development. It includes major project dependencies, such as specific technologies to be used, third-party vendors, development partners, and organizational constraints.</td></tr>
<tr><td colspan="3">Purpose: The System Vision is an entry point for the systematic analysis and modeling of functional and non-functional requirements. It forms the basic rational for further requirements and system design decisions, particularly, for determining Conditions, Scope and Limitations of the development. Finding the implicit assumptions of the different stakeholder/disciplines is the first step to identify often unnecessary preconditions and vague or conflicting requirements.

A rough context diagram describing the structural integration of the system into its operational environment (see Domain Model) supports the systematic identification of assumptions and dependencies.</td></tr>
<tr><td colspan="3">References: [Wie99] gives an overview of a vision specification and shows the role of a context diagram in domain analysis. [Gei05] outlines methodical steps of analyzing the operational environment and defining the logical system boundaries/interfaces.</td></tr>
</table>

Business Needs Artifacts (cont.)

Conditions and Scope		Mandatory
Responsible: ProdM	*Contributing:* RE, SA, Dev, ProjM, RelM	
Description: Determine the external considerations that influence system functionality and quality, and analyze their consequences for system design.		
General Conditions		Mandatory
General conditions are high-level design constraints that result from influencing factors such as design/development standards, social, economical, technical, contractual, organizational, or political factors. In general, they include functional and quality requirements that are crucial for the future acceptance of the product.		
Scope and Limitations		Mandatory
Determines the system-relevant parts of the application domain and environment. It outlines the relevant users and systems/components that interact with the system supported by a sketched context diagram (see *Domain Model*). Within product-line development this includes determining the scope and variations of initial release and subsequent releases.		
Purpose: Valid conditions are justified by *Business Objectives*, and must be refined and measurable specified by *Requirements* and *System Specification* artifacts. This includes the functional analysis of potential usage scenarios and architectural concepts.		
References: Overview description: [SS98], [LK95] provide an overview of sources for external factors that limit the functional system design. The Global Analysis approach in [HN+05] outlines methodical steps of analyzing external factors and concluding their impact on system design.		

Business Needs Artifacts (cont.)

Return of Investment (ROI) and Risk Analysis		Recommended
Responsible: ProdM	*Contributing:* RE, SA, ProjM, RelM, QA, Dev	
Description: ROI estimates/calculates the potential revenue of the new system in relation to its proposed development and life-cycle costs.		
ROI Calculation		Recommended
Requirements-related cost-benefit-analysis includes the assessment of long-term consequences, e.g. market success/failure or consequential costs of requirements/system design decisions.		
Business Risk Analysis		Optional
Business risk analysis identifies development risk factors, their potential severity, and proposes strategies for mitigation. Requirements-related risk analysis is focused on identifying and analyzing “bad”, missing, conflicting, or vague requirements, assessing their impact on system feasibility, cost and market success, and supporting appropriate requirements and design decisions. It includes corresponding impact analysis when changing requirements or design decisions.		
Risk Calculation		Optional
It estimates possibilities for market, technology and project failure. It assesses volatile/vague requirements, the achievability of quality goals, contracting/supplier guarantees and reliability commitments. It also analyzes organizational and project risks for missing time, effort and cost targets.		
Purpose: ROI and risk analysis is used for prioritizing and deciding about requirements and system concepts. It indicates feature benefits in relation to their risks, and therefore works as a driver for communicating and consolidating requirements and specification documents.		
References: Overview of ROI of software projects [Jon98] and risk analysis [Jon94]; [KK06] proposes an ROI approach that takes organizational factors into account; [Wie99] discusses issues of requirements related risk analysis.		

Business Needs Artifacts (cont.)

System Success Factors		Optional
Responsible: ProdM	*Contributing:* RE, SA, QA	
Description: Define the 'unique selling point' of the system to be developed and determine how success will be defined and measured for this system.		
Key Features and Requirements Describe the factors and key features that are likely to have the greatest impact on achieving product success. Establish measurable criteria for assessing whether the business objectives have been met. Examples of these criteria are market share, sales volume or revenue, customer satisfaction measures, transaction-processing volume and accuracy, and other quality attributes. Prioritize the key system features and requirements, and take into account business, system, and product-line strategy. This could include an incremental development strategy by appropriately assigning requirements to the different releases.		Mandatory
Purpose: The defined system success factors are used as the measurable criteria of the product development. In the case of product-line development, the prioritized requirements are input for the corresponding release planning of the future system (product-line families).		
References: Kano Modeling [Ka+84] outlines a customer-oriented approach of prioritizing requirements of product features.		

2.3 Requirements Specification Artifacts

Figure 5 gives an overview over the *Requirements Specification* artifacts and their content items.

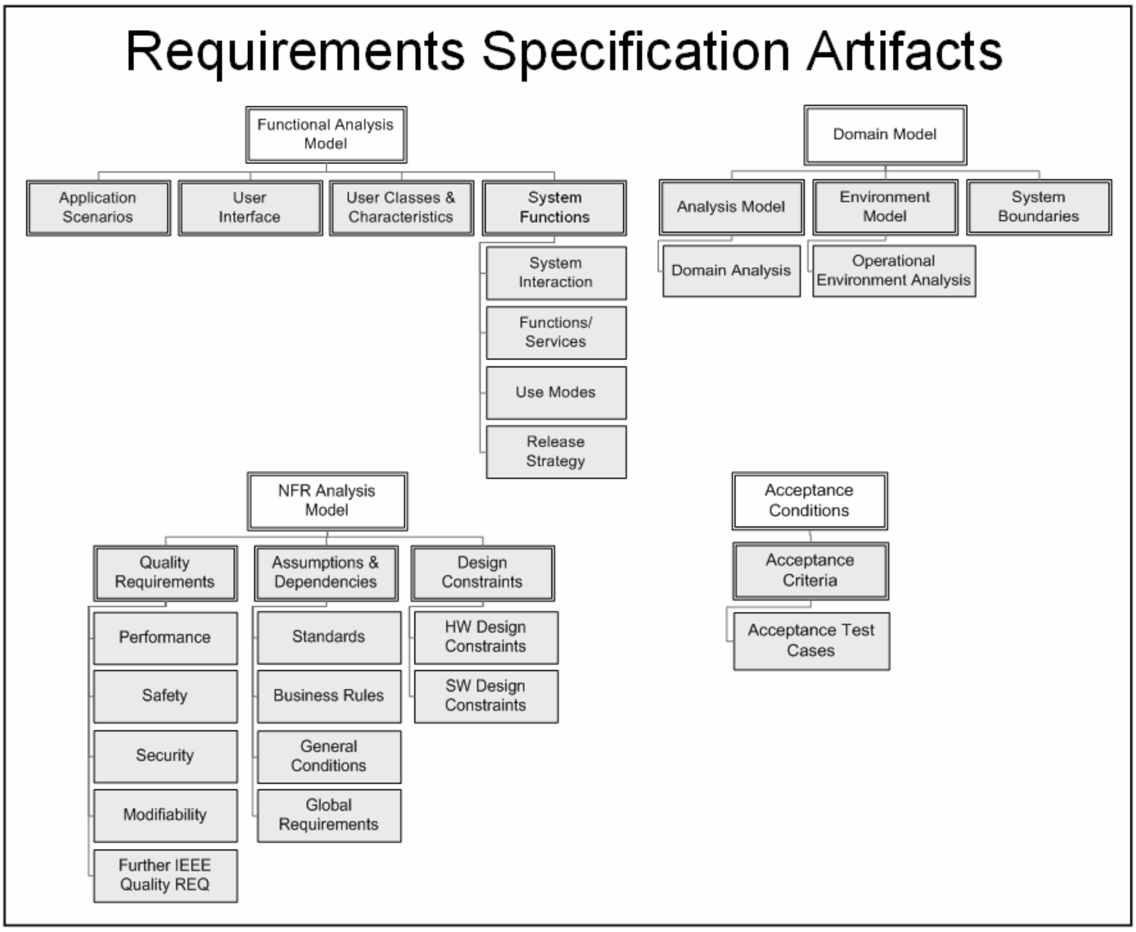


Figure 5: Overview of REM Requirements Specification Artifacts.

The *Requirements Specification* artifacts support the refinement, strengthening, and consolidation of the goals and high-level requirements of the Business Needs. The result is a precise specification of the functional and non-functional requirements of the system under consideration (SuC). The focus here is analyzing and structuring requirements and constraints from a customer and user perspective to guarantee the goal-oriented development of system solution concepts. The analysis models drive the rationale for system construction decisions and the traces to the business and product goals of the project. Therefore, these artifacts support the interdisciplinary communication and consolidation of goals and solution concepts in system development. The Requirements Specification artifacts include:

- *Functional Analysis Models:* They are used to analyze and model the business and application processes/scenarios, and to derive the required system services, their interaction behavior and quality attributes.

- *Domain Models:* They are used to analyze and specify the characteristics and variables of the given application domain and structure the operational environment of the SuC to define the scope and system boundaries of the future system.

- The *Non-functional Analysis Models:* They are used to analyze, structure and precisely define the quality goals, general conditions and other high-level requirements that constrain the functional design of the SuC.

These complementary analysis models lead to the overall specified *Acceptance Criteria* of the final system and its integration into the defined environment.

The Requirements Specification artifacts are described below.

Requirements Specification Artifacts

Functional Analysis Models		Mandatory
Responsible: RE	*Contributing:* ProdM, SA, Dev, QA, RelM	
Description: Analyze the usage of the SuC to derive and measurably specify required system services (system functions), boundaries, user interface and future system interfaces to the environment. This is carried out by describing/modeling application scenarios, system interaction, system use modes and refining and hierarchically structuring of system functions.		
Application Scenarios		Mandatory
Describe application processes in terms of structured business process and use case models. The scenario analysis is the basis for deriving user interface, system, and acceptance test specifications. Business process and scenario models identify and structure required system use cases/functions. Particularly, the scenario analysis supports the identification of assumptions about the system environment, the definition of system usage modes, pre and post conditions, and the description of representative system interaction patterns.		
User Classes and Characteristics		Recommended
Identify and analyze target user classes of the SuC and define corresponding user profiles by their characteristics and expectations of the system.		
User Interface		Recommended
Describes requirements and constraints on the interaction between the users and system in terms of dialog guidance and visible and physical input/output interfaces. This includes the description of default GUI standards, system (family) style guides or constraints to the interface design.		
System Functions		Mandatory
Define the services the SuC has to provide structured by major use modes, their subfunctions (according to the scenario analysis), and supporting communication functions to the system environment. The behavior of and interaction between the services (feature interaction) are analyzed and described by modeling usage processes, corresponding system interactions, and by deciding about critical use situations and system modes. This includes making appropriate assignments of task/control to users, components (of the environment) and the system services. In the case of product-lines and incremental development, additionally structure and specify required services/functions by variation models, and define the release strategy.		

Functional Analysis Models (cont.)
Purpose: These artifacts are used for communicating and consolidating customer/user requirements and defining the appropriate behavior and quality attributes of the system. Together with the *Domain Analysis Models*, they are the basis for constructing, validating and verifying the detailed specification of the system behavior (*Functional System Concept*). Particularly *Application Scenarios* and their derived *Behavior* and *Interaction Models* are the basis for specifying acceptance test scenarios (*Acceptance Conditions*). These functional descriptions are the basis for deriving prototypes to get feedback from potential system users (demonstration prototypes), system architects, or discipline-specific solution engineers (mechanical, electrical, software). They are major input for appropriate *User Documentation*. Customer-oriented behavior definition requires intense user/customer involvement in functional requirements analysis.
References: Business process modeling: LEU [DG96], ARIS [Sch02]; Use Case and scenario analysis are major tools for refining high-level non-functional requirements (*Non-functional Analysis Models*), and identifying and measurably specifying the quality requirements: [PH97], [DP+03]], [AR04, AF2], [Car95], [CL99], [PB+91], [HP05], [WP+98], [BS02]; User interaction and task analysis: [Som04], [Sut02], [Pat99]; UML description techniques: activity-, sequence-, use case- and state-diagram [UML]; Scenario-based evaluation of architectural design: ATAM [KK+00].

Requirements Specification Artifacts (cont.)

<table>
<tr><td colspan="2">Domain Models</td><td>Recommended</td></tr>
<tr><td>Responsible: RE</td><td colspan="2">Contributing: ProdM, SA, Dev</td></tr>
<tr><td colspan="3">Description: Specify the entities and characteristics of the application domain and define the operational environment of the SuC.</td></tr>
<tr><td colspan="2">Analysis Model</td><td>Mandatory</td></tr>
<tr><td colspan="3">This artifact includes causal and physical principles or business rules of the application domain, and it determines the relevant functions, variables, and constraints on the system design.</td></tr>
<tr><td colspan="2">Environment Model</td><td>Recommended</td></tr>
<tr><td colspan="3">Based on the Analysis Model, the Environment Model describes the operational environment of the SuC to determine the relevant components of the domain, their purpose, and information flow/interfaces to the SuC.</td></tr>
<tr><td colspan="2">System Boundaries</td><td>Recommended</td></tr>
<tr><td colspan="3">Using the Analysis and Environment Models, set the boundaries of the SuC and decide what is in and what is out of the current development project. Decide about the environmental constraints on the SuC, and illustrate using a context diagram.</td></tr>
<tr><td colspan="3">Purpose: Domain models are an interdisciplinary communication mechanism for understanding the application domain and defining the environmental constraints and interfaces to the SuC. They accurately specify the implications of the Scope and Limitation requirements and eventually cause the revision of pre-assigned design decisions.

The structural analysis of the system environment should be supported by data flow diagram/context diagram modeling.</td></tr>
<tr><td colspan="3">References: Domain analysis: [Jac95], Feature-oriented Domain Analysis (FODA) [KC+90]; Defining the system boundaries by data flow diagrams/context diagram modeling: [DeM79], [BP+99], [AG90], [Gei05].</td></tr>
</table>

Requirements Specification Artifacts (cont.)

Non-functional Analysis Models	Mandatory
Responsible: RE	*Contributing:* ProdM, SA, Dev, QA

Description: Derive and structure non-functional requirements from the high-level requirements and quality goals.

Quality Requirements	Mandatory

Structure measurable quality requirements by quality models. For each relevant quality goal, define specific quality factors, quality criteria, and corresponding metrics. In REM, measurable specification is supported by the *Functional System Concept* and *Design Constraints* artifacts.

Example quality goals/requirements are availability, reliability, performance, security, safety, usability, capacity, scalability, integrity, stability, and maintainability (portability, readability, modifiability).

Assumptions and Dependencies	Recommended

Analyze high-level requirements with respect to standards, business rules, global requirements, and general conditions to conclude functional requirements, quality requirements, and design constraints.

Design Constraints	Recommended

Derive and capture constraints on the hardware design (mechanical, electrical, deployment) and software design (platforms, protocols, frameworks, reuse of software, tools, programming languages, software architecture, coding standards).

Purpose: Understanding the implications of business goals and high-level constraints in context with the appropriate functional, quality requirements and detailed constraints is a pre-requisite for designing the best system solution.

Get further assessed, verified and completed by the functional system modeling of the *Functional System Concept* artifact and corresponding validation and verification techniques.

They constitute the traces to business and customer goals and allow comprehensive decisions about system design.

References: ISO 9126-1 Quality Model [ISO9126-1], Quality-Criteria-Metrics-Method [MR+77], quality model approaches: [SS98], [DK+05], [HP05], [BD+06]; goal refinement: [LD+98], [CN+00]; Global Analysis approach [HN+05]; Goal refinement and scenario-based evaluation of architectural design: ATAM [KK+00]; Feature-oriented Domain Analysis (FODA) [KC+90].

Requirements Specification Artifacts (cont.)

Acceptance Conditions	Mandatory
Responsible: ProdM	*Contributing:* RE, SA, QA, RelM
Description: Specification of measurable conditions and criteria for accepting the developed system before customer delivery.	
Acceptance Criteria Specify the acceptance of the final system by identifying test cases and measurement criteria derived from the *Business Needs* specification. It includes criteria for achieving domain specific interface and development standards. Test cases are developed from user scenario descriptions, and they verify that the system meets the required behavior and quality attributes.	Mandatory
Purpose: Acceptance conditions are used for customer acceptance criteria of the SuC. They are derived from the functional and non-functional *Analysis Models*. Specifying acceptance conditions forces all stakeholders to communicate and consolidate their own understanding about the development goals, the functionality and performance of the system solution.	
References: Approach to scenario analysis and its application to acceptance testing [HP+97].	

2.4 System Specification Artifacts

Figure 6 gives an overview of the System Specification artifacts.

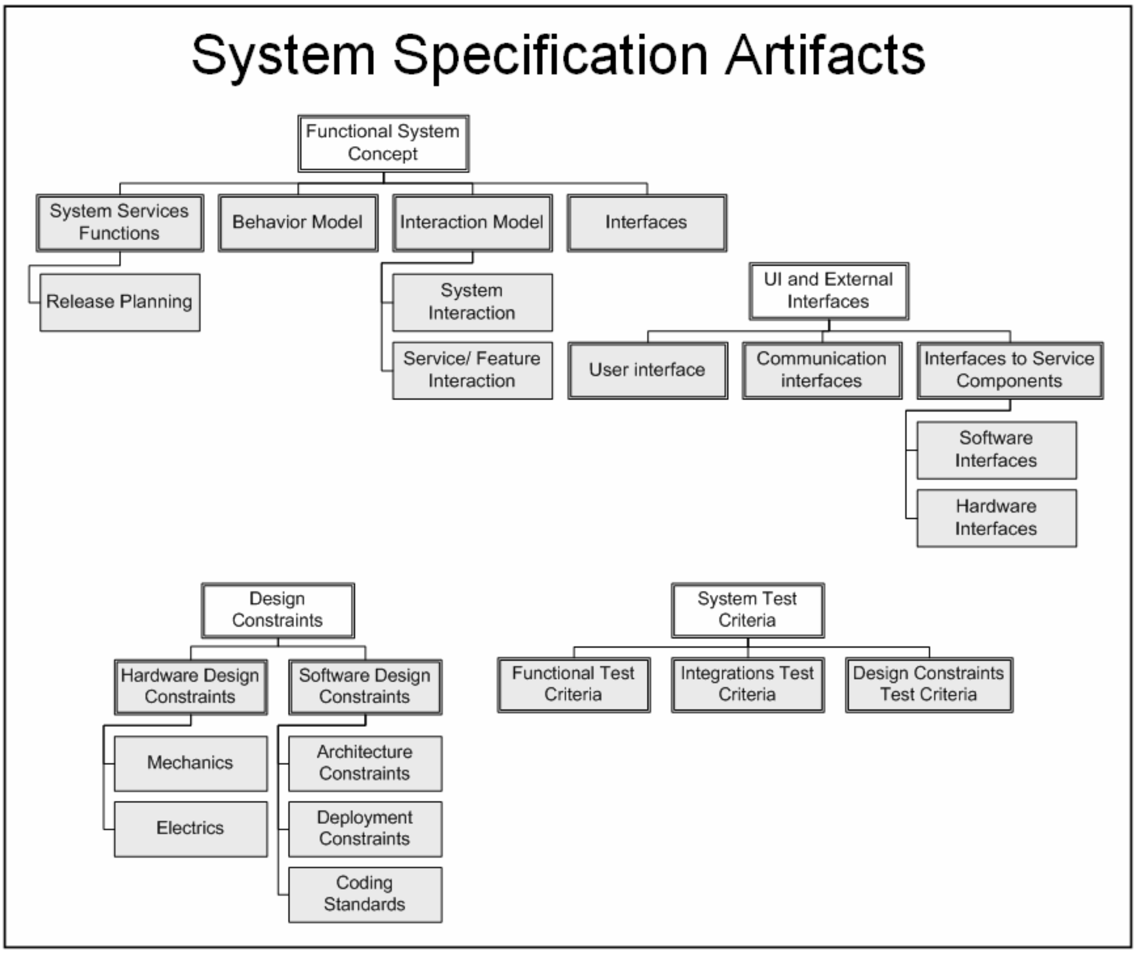


Figure 6: Overview of the System Specification Artifacts

The *System Specification* artifacts represent the developed and decided system solution concept. They specify the detailed system requirements and constraints, which are derived from and justified by the *Business Needs* and *Requirements Specification*.

The primary artifact of the *System Specification* group is the *Functional System Concept*. It specifies the required behavior of the system to be developed. Together with the *UI, External Interface* and *Design Constraints* definition, it forms a consistent and measurable basis for the realization of the entire system through the different engineering disciplines.

REM is built on a system modeling theory[2] developed by the TUM. It supports the structured description of the functional behavior of the systems in terms hierarchically structured system services (a kind of formalized feature trees) including modes/transitions and interaction modeling (*Behavior Model, Interaction Model, Inter-*

[2] The Focus approach [BS01]

faces). In addition, the concepts of hierarchically composed system architectures and state machines with input and output are provided.

Using system services within requirements engineering the functionality of a system can be built up from the beginning in a structured user-centered way. Thus all kinds of requirements and constraints can be captured and modeled by the corresponding functional views of systems.

The system modeling principles of REM can be addressed by informal to formal methods and description techniques. Using formal descriptions techniques allows comprehensive and automated verification and validation of requirements but it implies extra effort and special skills to do. This effort is not always justified or only for special components of the SuC e.g. safety critical components. Using textual, informal or semi-formal description techniques still allow the use of the underlying mathematical model for verification and validation using review technique for checking e.g. consistency and completeness rules. REM supports the combined use of textual to formal description techniques.

Functional system views are a major tool for verification and validation. Derived from the *Requirements Specification* products, missing or conflicting requirements in this models can be uncovered by the modeling and consistency rules and further be analyzed, validated and completed by the traces to the user/business goals of the project. This underlying methodical concepts and methodological instances of using REM are described in Chapter 4.

System Specification Artifacts

Functional System Concept		Mandatory
Responsible: SA	*Contributing:* RE, ProdM, Dev, ProjM, QA, RelM	
Description: Describe the system solution concept by specifying the offered services/functions of the system, their behavior, interrelations/dependencies, and their interfaces and interaction with the environment.		

System Services and Functions	Mandatory

The System Services and Functions describe and specify the finally determined system services in a structured way. The specification is defined within the specific product or product-line environment.

In general, a service is specified by the purpose of using the service and syntactic (data) and semantic (behavior) interfaces. The syntactical interface contains the types of messages, which can be exchanged with the service (see *Interfaces*). The semantic interface describes the required behavior of the service by relating valid sequences of input messages to sets of sequences of output messages. This behavior best is specified by the complementary techniques of *Interaction* and *Behavior Modeling.*

In the context of the concrete system development, the service specification has to be complemented by defining the specific assumptions about the environment (pre- and post conditions of the overall system states). Particularly purpose, pre- and post- condition determine the specific use of the service and define the integration of the system into the product-specific system environment

This use(r)-centered specification includes structuring the services into “original” *application services* and supporting *communication services* to other systems/components of the environment, as well as ordering them into “is part” hierarchies – several system services realize/specialize one required application or communication service. It is complemented by the *UI* and *External Interface* specification. .

Behavior Models	Mandatory

Specify the required system behavior by user modes, system/service states, and corresponding transitions. The state of a system or service can be changed via input from the environment (user or other systems) or by interacting with other services.

These models describe the comprehensive behavior of the system. They help formulate system design decisions, which get evaluated, possibly revised, and completed during *Scenario* and *Interaction Modeling*.

State machines described by state transition diagrams or state transition tables are effective methods for specifying behavior in terms of state transition rules. In a nutshell, a rule for transitions from one state to another comprises a precondition that works as a guard and determines under which conditions a transition may be executed; input data - the input required for the transition; output data - the output generated by the transition, and a post condition characterizing the state the system/service must hold after the execution of the transition.

Functional System Concept (cont.)	
Interaction Models	Recommended
Specify system/service behavior by: • interaction patterns of the system with the environment • interaction patterns between two or more system services, if they are interrelated (*feature interaction*). They may mutually depend on each other (one service uses/is used or controls another service). These sets of interaction patterns specify the required data/message exchange and can be systematically completed by adding constraints (pre- and post- conditions on data or system/service states). The objective here is to exclude unwanted system behavior or feature interaction.	
Interfaces	Mandatory
Specify the logical data interface definitions of the system/services. These *Interface Specifications* describe the types of messages that can be exchanged with the environment in terms of their data types and structure definitions. These can be status information messages about physical devices or processes, or messages to control or communicate with external systems or services. This interface specification is complemented by properties of the exchanged message streams including constraints of the product-specific system environment, the specified *External* logical and physical *Interfaces*.	
Purpose: The *Functional System Concept* describes the agreed to and precisely specified design of the system. Central techniques to develop and evaluate the system design are *Scenario*- and *Interaction Modeling*. They identify the required application and interaction steps, the involved components and services, and systematically define the necessary data exchange and system states. Quality requirements can also be refined and measurably specified using these modeling techniques. Together with the traces to the business and user needs, the functional system concept is communication base for the further development and integration within the different realization disciplines (mechanics, electrics, software). Consistency rules between the different specifications or system views allow verification and validation, e.g.: a messages/data described in a scenario interaction must be specified in the interface definition and in the corresponding behavior specification. An interaction sequence must be incorporated in the state-based behavior specification. A state-based service specification can be clearly described by a corresponding set of “typical” interaction sequences (see also Appendix C, Figure C-4). These functional models are basis for building prototypes and simulations of the developed system concept.	
References: Semi-formal to formal approaches of system modeling: UML 2.0 [UML], 4+1 Views Model [Kru95]; Open source System Modeling Language (SySML) [SySML]; Use Case and system state modeling of the QUASAR approach [DP03], traditional structured analysis concepts like [DeM79], SSADM [AG90], SA/RT [MP84]; mathematically founded system modeling in FOCUS [BS01] and AutoFocus [AutoFo-	

cus, AF2], [Sch04].

System Specification Artifacts (cont.)

<table>
<tr><td colspan="2">User Interface and External Interfaces</td><td>Recommended</td></tr>
<tr><td>Responsible: SA</td><td colspan="2">Contributing: RE, Dev, ProdM, RelM, ProjM</td></tr>
<tr><td colspan="3">Description: Specify the interfaces of various users and components of the environment that interact with the system according to the overall system design.</td></tr>
<tr><td colspan="2">User Interface</td><td>Mandatory</td></tr>
<tr><td colspan="3">Defines the data, behavior and physical interfaces to the users of the system including the various modes of user interaction. It includes the specification of standard user interface components (see Interfaces of Service Components). The behavior specification is derived from the corresponding functional service specification of the system.

The User Documentation is written from the Functional and UI Specifications, and usually is developed in parallel.</td></tr>
<tr><td colspan="2">Communication Interfaces</td><td>Mandatory</td></tr>
<tr><td colspan="3">Define the communication protocols and physical interfaces of the components of the system environment that the SuC is communicating with, controlling, or gets controlled by. These interfaces specification is consolidated by all involved solution/engineering disciplines.</td></tr>
<tr><td colspan="2">Interfaces of Service Components</td><td>Mandatory</td></tr>
<tr><td colspan="3">Specify the communication protocols (data and interaction) and physical interfaces of the service components that the system is using.</td></tr>
<tr><td colspan="3">Purpose: These artifacts complement the functional system design (Functional System Concept), and they specify the logical and physical integration into the designated system environment. It’s the result of design decisions and represents the (communication) synchronization interfaces between all involved engineering disciplines.</td></tr>
<tr><td colspan="3">References: User interaction and interface modeling: [Som04], [Sut02], [Pat99].</td></tr>
</table>

System Specification Artifacts (cont.)

Design Constraints	Mandatory
Responsible: SA	*Contributing:* RE, Dev, RelM, ProjM, QA
Description: Specify the constraints on the design and realization of the envisioned product or product-line within the involved engineering disciplines.	
Hardware Design Constraints Specify functionality and interfaces including constraints on anticipated functional and physical design; e.g. the usage of specific hardware, standard devices, or procedures.	Recommended
Software Design Constraints As an addition to the *Functional System Specification*, specify the decided constraints on software design and realization. This includes platforms, frameworks, programming languages, architectural constraints, deployment constraints, coding standards, and constraints of tools to be used.	Mandatory
Purpose: Restrict the further design and implementation to make sure that specific strategies on hardware and software are followed.	
References: Examples of reference architectures are provided by [SUNREF] (infra-structure), [QuasarREF] (application), [IBMREF] (business-pattern); Technical plat-forms: [J2EE] (sun java-plattform), [.NET] (.net platform); Style guides: [JavaStyle], [C++Style], [GNUStyle].	

System Specification Artifacts (cont.)

System Test Criteria		Mandatory
Responsible: SA	*Contributing:* Dev, QA, ProjM, RelM, RE	
Description: Describe the tests and review criteria for the verification of the requirements.		
Functional Test Criteria Provide a suite of functional test cases that have to be passed to validate that the system fulfills the functional requirements.		Mandatory
Integration Test Criteria Provide a suite of integration test cases that have to be passed to validate that the implemented system fulfills the architectural requirements.		Mandatory
Design Constraint Test Criteria Define the reviews that have to be carried out to verify that the implemented system meets the documented design constraints.		Recommended
Purpose: System test criteria are used to help ensure that the implemented system fulfills the specified requirements.		
References: UML2 Testing Profile [UML2Test]; TTCN-3 European Telecommunications Standard [TTCM Test].		

Chapter 3 Process Strategy and Tailoring

REM does not enforce a specific development process. It can be adapted to any system and software life-cycle model. REM gains this flexibility from its tailoring concept very much along the lines of the V-Model XT (see [VM-XT]). REM provides basically a model of RE work products – the RE Artifact Model - with a variety of instances and process models to be selected. The artifact model defines the basis for choosing the appropriate methods and techniques.

The basic steps for tailoring to specific project instances are:

1. *Prune the artifact model*: Cut and tune RE specification artifacts and their content.

2. *Fix the document structure*: Define the document structure and content.

3. *Choose methods*: Select methods and description techniques.

4. *Define process*: Define the process by the desired decision gates.

5. *Specify Roles*: Define roles and responsibilities.

RE artifacts are labeled in REM with the following attributes: mandatory, recommended, and optional. Mandatory artifacts must be part of any tailoring instance.

3.1 Artifact-oriented Process Support

The REM artifact model is the basis for deciding about methods, tailoring, and process definition.

3.1.1 Milestones and Quality Gates

REM defines milestones and quality gates in terms of the completion levels of the RE artifacts. Figures 7 and 8 show this concept of artifact-oriented process definition:

The artifact model defines the content and structure of the overall specification documents, and measurably defined completion levels of the specification documents form the basis for reviews and decision making in product definition and realization.

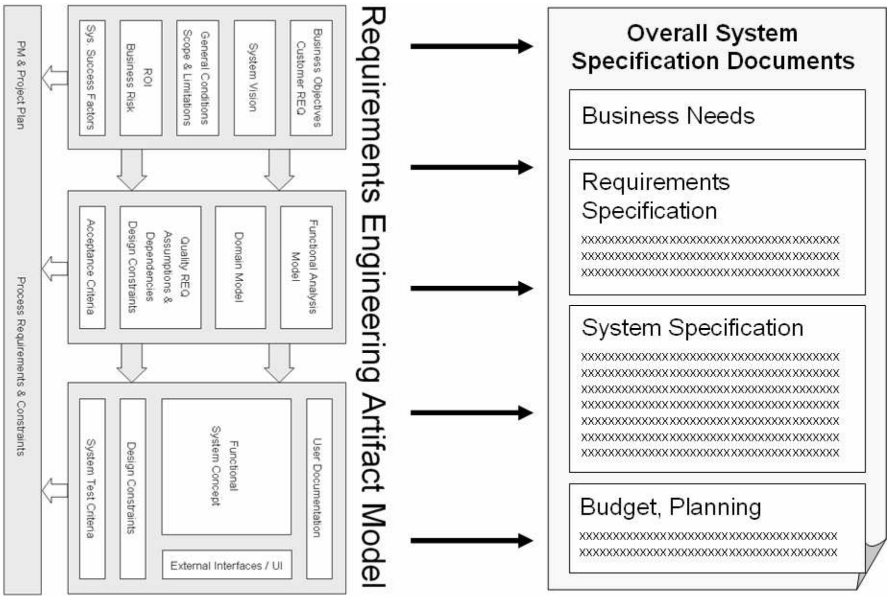


Figure 7: RE artifact model defines the content and structure of specification documents.

Figure 8 shows how to define decision gates by the completion levels of the RE artifacts. For simplicity, these completion levels are indicated as percentages of the RE artifacts across system development. The percentages represent the desired degree of quality and completion of the artifacts at the time of the decision gate.

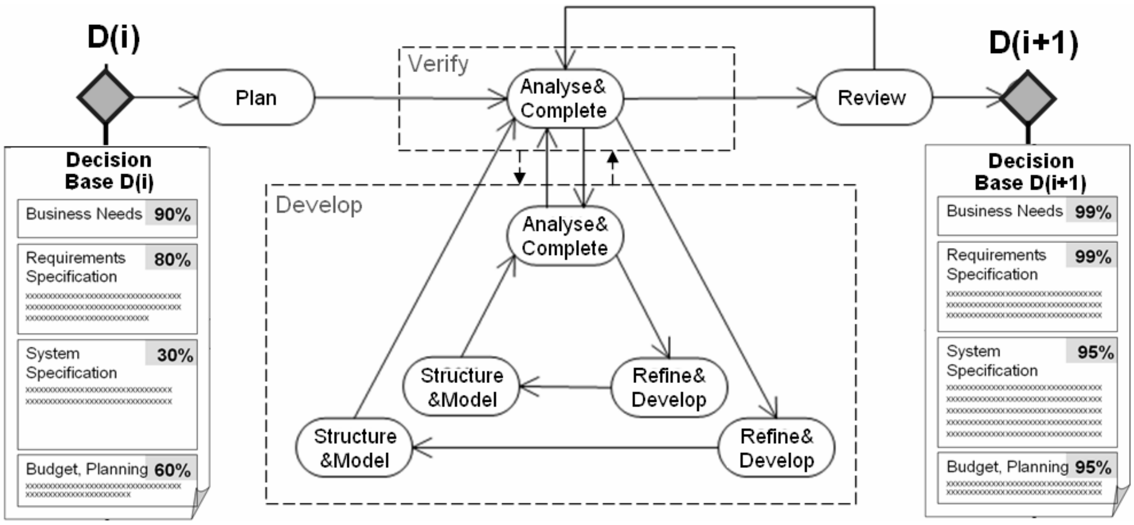


Figure 8: Process definition by the completion levels of specification documents.

Corresponding versions of the specification documents form the *decision base D(i)* at specific milestones or quality gates. Additionally, they are inputs for *artifact-specific RE control loops* in the project.

Decision gates can be the milestones and quality gates of Product Lifecycle Management processes (PLM) like the Siemens PLM. The number of quality gates in concrete projects depends on the desired process approach; e.g., agile, component-oriented, traditional.

Artifact-Specific RE Control Loops

The process definition approach in Figure 8 shows a pattern of RE control loops that supports iterative requirements development. The *RE modeling cycle* shown in Figure 9 is characterized by alternately *developing* and *verifying* requirements. Every cycle is completed by an interdisciplinary *verification* activity to evaluate and consolidate the so far *developed* requirements and system concepts. Conducting the *verification* and deciding about the next activities are based on the tailored desired quality levels of the RE artifacts. This *verification* is conducted by formal reviews, inspections, or automated tool supported verification and validation methods. The control loop starts after *decision gate D(i)*, where - according to the predefined quality level or versions of the specification documents - requirements are completed and reviewed.

In Figure 8, *Plan* summarizes the activities to define the specific work packages; i.e., the steps and activities to develop the specifications that are required for decision base *D(i+1*). In addition to organizational, financial and timing conditions, the major input to the planning activity is the set of required documents, their required content structure (tailored parts of the RE artifact model) and related roles. A final *review* is conducted before the decision about the project/product can be made at decision gate *D(i+1).*

3.1.2 RE Modeling Cycle

Figure 9 shows the iterative RE modeling cycle of analyzing and completing requirements based on the fundamental structuring and modeling concepts of the REM (which are outlined in Chapter 4). RE is an iterative task of integrated requirements and system specification, where initial models have to be built, reviewed and completed in recurring cycles. They consist of the activities *Analysis, Refinement, Classifying* and *Modeling*, as explained below.

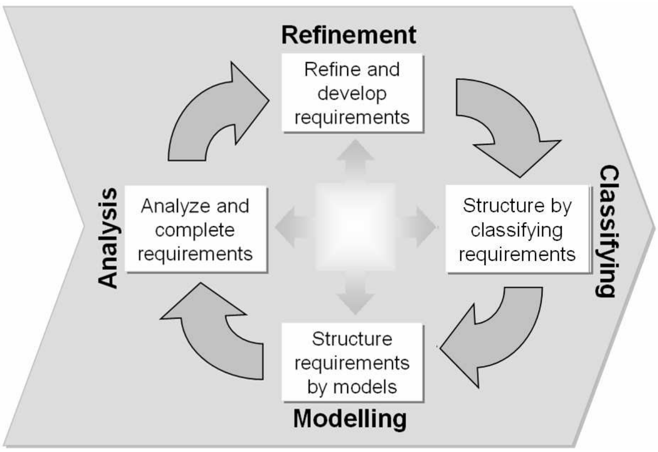


Figure 9: Iterative RE cycle of analyzing and completing requirements

Analysis – Analyze and complete requirements.

Analyzing includes the activities to understand and communicate requirements. It starts with refinement and structuring of collected requirements and ends with consolidation and completing of the so far modeled requirements and system concepts.

Refinement - refine and develop requirements.

The main task of understanding and communicating requirements is the structuring and developing the problem or solution specifications; i.e., to model, refine and develop new requirements along domain-specific patterns, which are the result of tailoring the RE artifact model.

Classification – structure by classifying requirements.

The first step of modeling requirements is to classify them according to the underlying structuring concept of the relevant specification templates.

Modeling – structure by modeling requirements.

Modeling in REM means the refinement and classification of requirements and the stepwise construction of functional solutions using the modeling concepts of REM (see chapter 4). These concepts allow the integrated development and verification/validation of requirements and solution models by using the tracing and consistency rules of the RE artifact model and their consolidation and completion in further RE cycles. Text, visualization techniques, or formal modeling languages can describe requirements and system models.

Structuring and modeling is a prerequisite for completing requirements specifications, as it is a recommended approach for discovering and showing vague, missing or conflicting requirements. It allows the communication and consolidation of specifications according to the business and customer needs of the system development. The structuring and modeling rules of REM allow the use of formal review techniques, and in the case of formal system modeling, one may use formal completeness and consistency proofs. The appropriate modeling approach is selected according to the specific project and domain needs.

3.2 Tailoring

For the tailoring steps of REM, each concrete project instance consists of:

- a subset of the REM RE artifacts,
- for each artifact, a subset of its content items that are selected,
- a definition of a document structure for the artifacts and main content items together with a definition of completion levels,
- a method to develop the artifact, and
- a process that defines the development sequence and delivery date for each version of the artifact (completion-levels).

With tailoring, we adapt and complement REM to the needs of an individual product, system, or project. In addition, project team members are assigned roles for accomplishing the RE activities.

3.2.1 Artifact Model

For tailoring the REM artifact model, the REM artifacts as well as their main content items are selected that are relevant for the project under consideration. We distinguish three classifications of RE artifacts and main content items.

- **Mandatory** artifacts and content items are absolutely required for a REM compliant tailored RE process.
- **Recommended** artifacts and content items can be eliminated by tailoring, but then a rationale is required as part of the tailoring process.
- **Optional** artifacts and content items are not absolutely required for a REM compliant RE process, but the decision to eliminate them should be justified.

If an RE artifact is classified as *mandatory*, its *recommended* or *optional* content items can be omitted. If a *recommended* or *optional* artifact is selected, its *mandatory* content items cannot be deleted.

The result of tailoring can be a reduced set of RE artifacts and content items. But then a rationale of why certain parts have been eliminated is recorded in a corresponding project tailoring document.

3.2.2 Document Structure

The RE artifacts can, but not necessarily, determine the document structure of an REM compliant RE process. Some artifacts can be combined into one document while others may be represented by several documents. Here documents include a wide variety of presentation formats including mixtures of text, graphics, data base content, tool output, software, screen shots, etc.

The result of tailoring REM in terms of a document structure is a list of documents with their relationships to the tailored set of RE artifacts and content items.

3.2.3 Methods

For each artifact or content item it is determined which methods and tools are used to produce those results and by which techniques the results are documented. This also includes methods for quality assurance such as reviews or consistency analysis.

REM itself does not define new methods. Rather, it represents a framework that summarizes various established or promising emerging techniques for requirements analysis and system specification. These vary from informal text-based methods and semi-formal description techniques like UML to mathematically founded formal approaches as they are developed and have been successfully used at the TUM. Chapter 4 outlines some analysis and validation methods that are supported at TUM.

3.2.4 Process

Figure 10 summarizes the tailoring concept of REM. *Activities*, milestones (*Decision Gates*) and *roles* are structured in terms of *RE artifacts* or respectively determined specification *documents*. Tailoring is done by pruning the artifact model to domain-/project-specific *RE artifacts*, determining specification *documents* and *completion-levels* to define the types and number of *decision gates*[3]. It includes determining *Methods* and *tools* for constructing the tailored RE artifacts and documents.

The result of the design of the process instance is a work plan with activities to develop the documents and RE artifacts including their content items. Corresponding to the defined *Completion Levels* of the RE artifacts, the documents may be developed iteratively as draft versions.

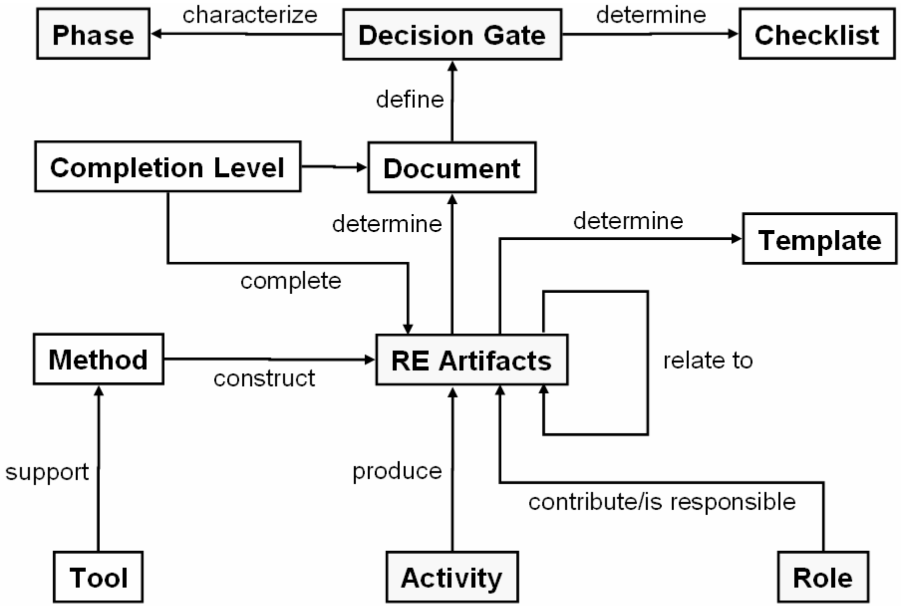


Figure 10: Overview of the tailoring concept of REM.

This artifact-centered tailoring of REM defines the basis for quality and progress measurement of the RE process by providing templates, checklists, methods, and tools. Thus, REM supports the business need for the effective development of requirements and system definition.

Figure 11 gives an example of tailoring REM to the Siemens product life cycle process (PLM) and the corresponding completion level of the specification documents at the PLM milestones M100, M1500, and M200. Figure 11-1, 11-2 and 11-3 show the corresponding parts of the RE artifact model. The actual specification documents at the decision gates are used for product and project decisions.

[3] This depends on the decided process strategy; e.g., traditional, agile, component-oriented.

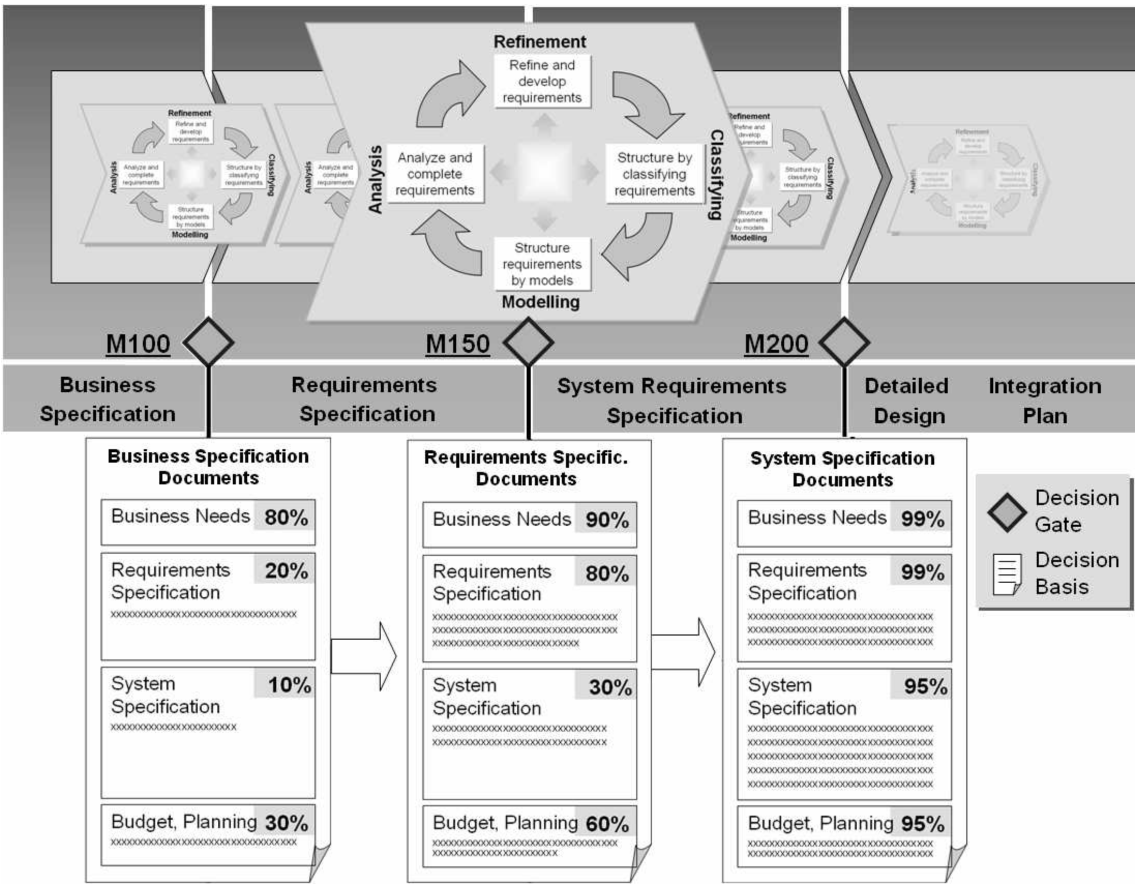


Figure 11: Decision gates defined by completion levels of specification documents.

As shown in Figure 11, 11-1-3, the specification documents at the decision gates M100, M150, and M200 can be described by the following documents one typically finds in current Siemens development practice:

The *Business Needs* artifacts form a *vision and scope document* as the basis for decisions about the further specification and development of the product at decision gate D0 - the first major project milestone after market-oriented product goal definition.

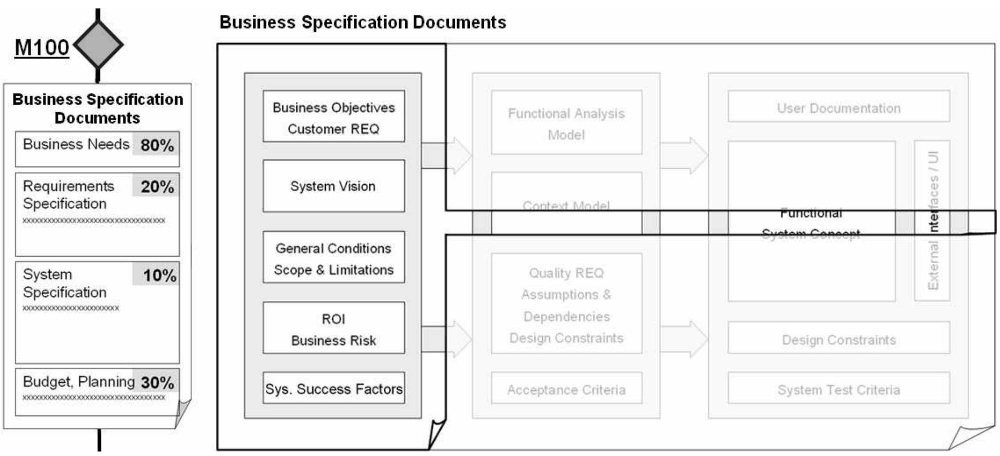


Figure 11-1: Completion level for Business Specification Documents.

The customer-oriented *Requirements Specification* documents include the Business Needs specification and promising system concepts. They form the decision base at D1 and correspond to the documents typically found in Siemens Germany, usually called "Lastenheft".

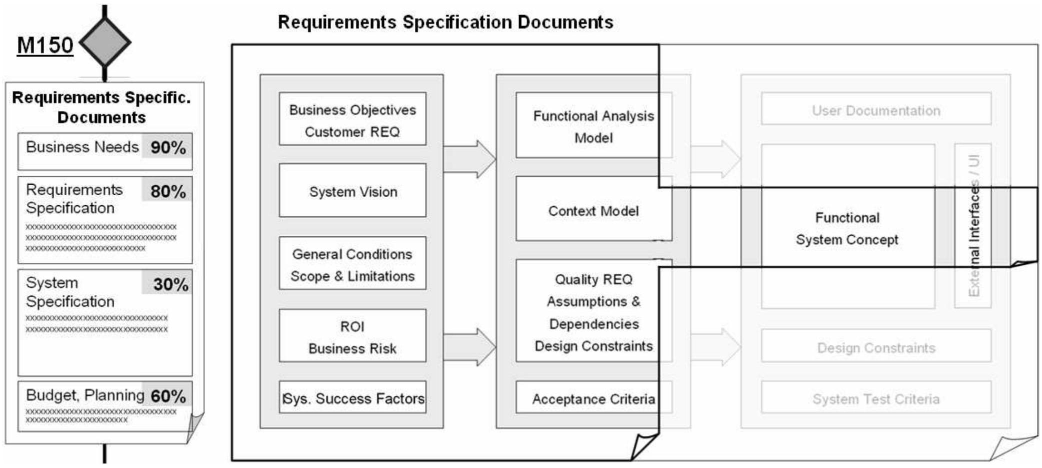


Figure 11-2: Completion level for Requirements Specification Documents.

The *System Specification* documents are the final result of multiple iterations and the consolidation of customer and domain requirements with different system options. The finally decided system concept and its specified requirements must be justified by business goals and therefore, the documents include the agreed *Business Needs* and *Requirements Specification* artifacts. The *System Specification* documents form the decision basis at the D2 decision review, and they correspond to the documents in Siemens Germany usually called "Pflichtenheft".

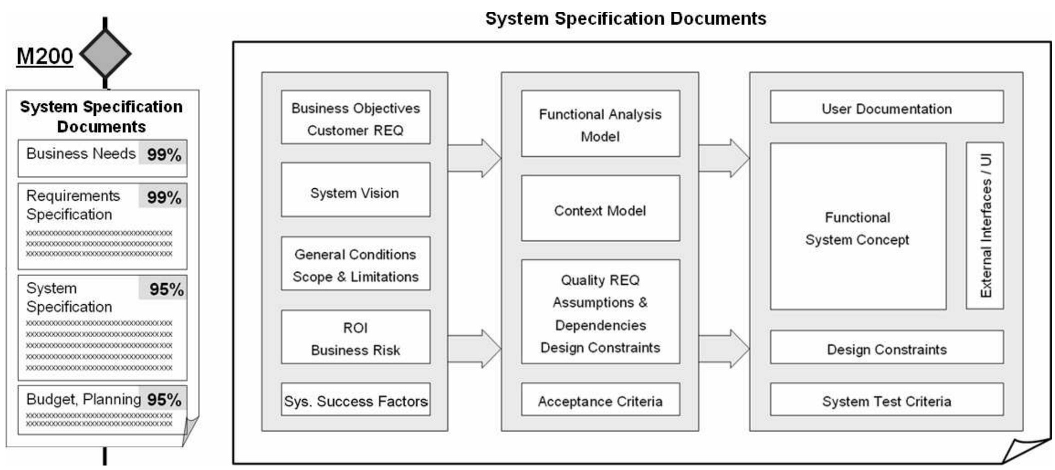


Figure 11-3: Completion level for System Specification Documents.

Selecting a more iterative or agile process usually means more decision gates and a more incremental completion of the documents and related RE artifacts. In an agile process one would sometimes start the implementation phase earlier and carry out the implementation in parallel before the *Requirement Specification* is frozen (concept of requirements log).

3.2.5 Roles

Finally, to complete the tailoring and instantiation of REM for an individual project we have to assign team members to the roles. Since the roles are connected to RE artifacts, for the case when a document comprises several artifacts with different roles being responsible, a single individual is chosen as responsible for the document ownership.

3.2.6 Result of Tailoring and Instantiation

In summary, the result of tailoring is a project plan consisting of:

- a set of RE artifacts including their selected content items,
- a document structure where each document's content is defined by the included RE artifacts or content items,
- an assignment of methods and description techniques for each RE artifact or content item,
- a concrete process with activities for developing the documents, and
- an assignment of team members to roles and roles to documents, RE artifacts, and content items.

These five steps are used to tailor REM for an individual project.

Like for the V-Model XT partial tailoring and instantiation of REM leads to restricted reference models that are REM compliant and can be used for a specific class of projects, products and systems. This can be used to design domain specific meta-models for RE complaint with REM.

Chapter 4 Analysis Views

In this chapter, we show how REM supports analyzing and validating requirements. The methods described are from the AutoRaid/AutoFocus research results [AR04, AutoFocus, AF2]. They are essentially laid down in the structure of the RE Artifact Model, and they enable measurable progress and quality control of system development.

4.1 Requirements Classification, Structure of Artifact Model

REM defines the relationships and dependencies among basic classes of requirements. The RE artifact model is structured according to the following principles:

- a goal-oriented classification of requirements,
- an approach to functional system modeling; i.e., modeling functional requirements by functional system views.

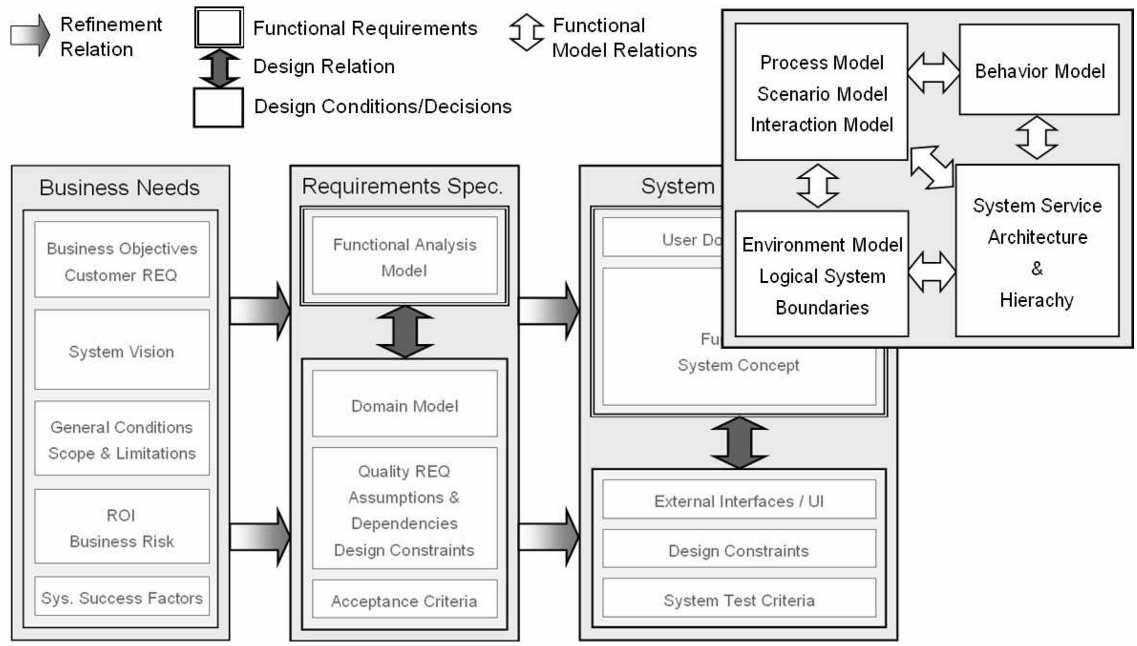


Figure 12: Requirements Classes and Dependencies of the RE Artifact Model

An overview of the goal-oriented classification and functional modeling concepts is given in Figure 12. This structure of the artifact model defines core dependencies between the different kinds and abstraction levels of requirements. They form analysis and construction rules for requirements specifications and allow the measurable verification and validation of derived requirements and system concepts. By comparing the content of specifications against these rules, missing and conflicting requirements can be found and corrected. Therefore, these structuring rules of the artifact model provide a measurement basis for progress and quality control in system development.

A brief description of these structuring concepts and their use for analyzing, verifying and improving requirements specifications is given below.

One methodological instance of this integrated RE and system modeling concept of REM is given in Appendix C by the data model of AutoRAID/AutoFocus. This tool provides a prototype realization of REM based on a mathematically well-founded system modeling approach.

Goal-oriented Classification of Requirements

The goal-oriented classification structures requirements into:

- the three abstraction-levels of *Business Needs*, *Requirements Specification* and *System Specification* with their *Refinement* dependencies (*Relations*) and

- the *Design Relationship* that distinguishes between *Functional Requirements* and *Design Conditions* in the *Requirements* and *System Specification*. Every *Design Condition* represents previously made *Design Decisions*, and *constrains* the further refinement and design of the *Functional Requirements*.

Requirements additionally specified by general attributes like “requested by”, rational, status (suggested, accepted, rejected …) and priority, this structure can be used to check the quality of the requirements.

Business Needs include business and product goals like portfolio decisions, and major functional or quality requirements of the application domain or system. They *justify* the refinement and design of the requirements in the *Requirements* and *System Specifications*: Every agreed to *Business Need* has to be implemented and measurably specified by one or more requirements of the *System Specification.* For every requirement in the *Requirements* and *System Specification,* a *Business Need* must exist that justifies its specification/design decision to which it can be traced.

The *Functional Requirements* of the *Requirements Specification* analyze and specify the usage functionality[4] of the system under consideration. They have to be further analyzed and designed using the functional modeling views/techniques that underlie the *System Specification* RE artifacts.

The *Design Conditions* of the *Requirements Specification* such as quality requirements, standards and further global constraints determine the “high-level” non-functional requirements that *provide* and *restrict* the refinement and design of the Functional Requirements. They also have to be analyzed, refined and specified by the functional system and constraints modeling of the *System Specification* RE artifacts. Some techniques to refine these high-level non-functional requirements are outlined in section 4.2. Every specified *Design Condition* represents previously made *Design*

[4] Functionality that is offered to users as both humans or other systems

Decisions. They must *be formally agreed* to by corresponding interdisciplinary reviews and management decisions, and justified by the *Business Needs*.

Functional Requirements of the *System Specification* are part of the designed and decided functional system concept. They have to be measurably specified by functional system models. Also, they must *be formally agreed* to by corresponding interdisciplinary reviews and management decisions, and justified by *Business Needs*. Together with the specified external interfaces and design constraints of the *System Design*, they define the agreed to and complete system specification.

The *Design Conditions* of the *System Specification* define the agreed to interface specification of the system environment and determine the formally decided constraints for further development within the different engineering disciplines. Therefore, they *provide* and *restrict* the further design and implementation of the functional system design. Also, they must *be formally agreed* to by corresponding interdisciplinary reviews and management decisions and justified by *Business Needs*.

Analysis and Modeling by Functional System Views

The RE artifact model underlies a general concept of how to describe systems and their required behavior. It can be described by semi-formal description techniques like UML [UML] and mathematically well-founded modeling approaches like AutoFocus [AutoFocus, AF2]. This functional modeling concept refines and structures the system requirements into the required *services* of the system and the system modeling views of:

- *Process Modeling, Interaction Modeling, Behavior Modeling, Structure Modeling (Environment* and *System Boundaries)* and *Data Modeling* with their

- *Model Relations* (consistency dependencies).

These functional modeling views cover the *Functional Requirements* and detailed *System Specification* of the RE artifact model. They are primary techniques to *refine* the high-level goals/requirements and construct, design, and verify system solution concepts. The model dependencies between the complementary system modeling views define consistency constraints, which can be used to check the quality of the system specification.

The *Process View* describes the process and scenario modeling of business and system use processes. The focus is on analyzing and defining the main application tasks and their logical interaction with the overall system and the system under consideration. Together with the complementary modeling views, the main *services* or *system functions* of the future system and their hierarchical ordering are derived and specified.

The Structural View is used to model the relevant actors, components, and their relationships. It models the logical structure of the application domain such as the operational system environment or corresponding product-line components. The system boundaries will be derived and the required interfaces defined. In this way a logical architecture is defined.

The *Interaction View* models the communication between the system and its environment (actors, components) while executing the application processes and system services. This modeling technique is used to analyze and derive the system behavior, the necessary system data and specify the system interfaces (see the *Process View* and *Process Analysis* above).

The *Behavior View* models the required system behavior in terms of states and transitions. It is used to analyze modes of system use and failure critical scenarios, and to derive a behavior specification of the required system.

The *Data View* specifies the overall data of the system. Major parts of this view are the interfaces, data structure definitions, and the state-defining variables and constraints of the system.

Using the Construction Rules of the Underlying System Concept

By using view-based modeling of system requirements with a common system concept, a set of consistency constraints is defined. It defines how the model elements of the different system views are related and how the consistency rules complement view-based specifications; e.g., messages/data described in a scenario interaction (*Interaction View*) with the system must be specified in the system interface specification (*Structural View*). Figure C-4 in Appendix C shows an instance of a system model and its construction rules; i.e., the data model of the system specification approach used in AutoFocus.

These rules can be systematically used to verify the consistency and completeness of the specified requirements and system models. If conflicting and missing requirements are discovered, they can be systematically treated and resolved by tracing them to the specified business and user demand (the *Business Needs* and *Design Condition* requirements of the artifact model). This tracing enables negotiation and completion of requirements according to the business needs of the actual system development.

4.2 Support for Analysis and Completion of Requirements

The RE artifact model defines the underlying content and structure of the overall requirements and system specification documents of a development project. As shown before, such a structure defines core modeling and consistency rules between different aspects of requirements specification. These construction and consistency rules provide fundamental and specific methods for verification/validation and customer-oriented development of system concepts.

Figure 13 summarizes the various analysis views on structured specification documents that are provided by the REM approach. An analysis view describes a specific aspect under which a model or specification is evaluated or analyzed. In REM, they can be distinguished into *basic* or *advanced/method specific Analysis Views*.

The methodical use of this analysis approach is demonstrated by the specification examples conducted at the TUM based on the AutoRAID/AutoFocus method and tool [GG+06, Sch04].

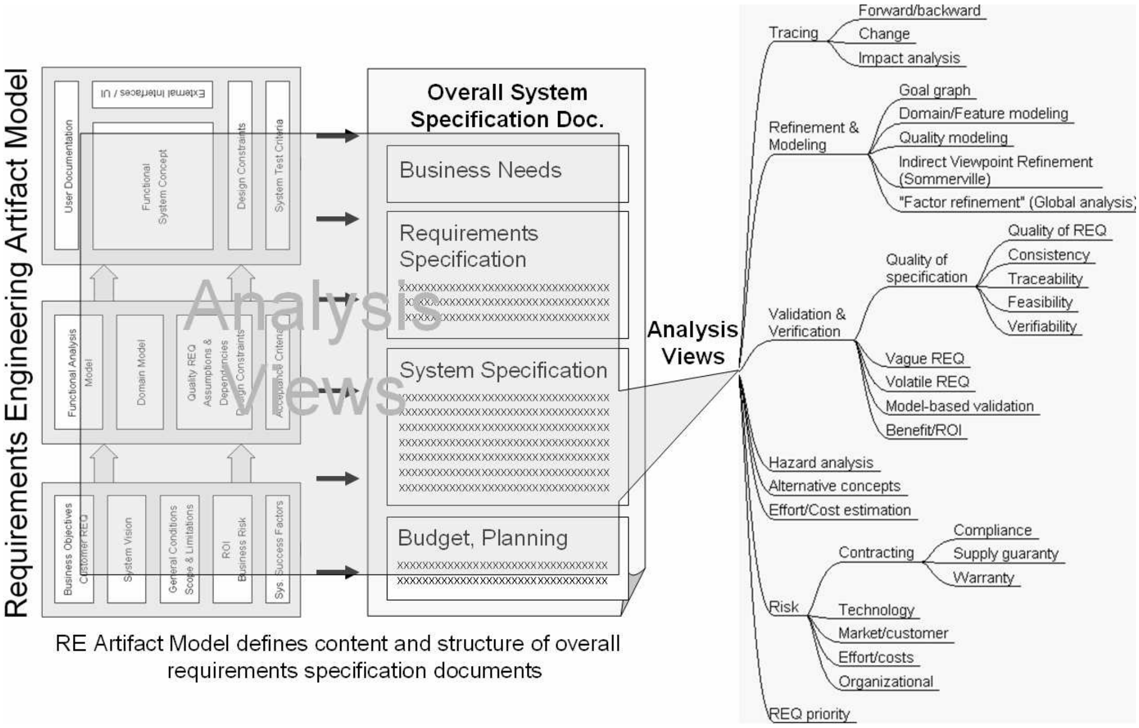


Figure 13: Supported Analysis Views

Basic Analysis Views

Basic analysis views are general evaluation aspects of requirements that are supported by REM for all types of development projects. Corresponding methods use the consistency rules of REM and can be applied using various description styles - from informal to formal:

- *General tracing* analysis based on the specified model links between requirements. Change and impact analysis can be conducted.

- *Validation and verification* support

  In addition to basic informal linguistic methods of text review such as found in [Rup02], REM supports quality and completeness analysis of requirements and specification documents. General attributes of requirements like rational, priority or status can be extended with RE artifact model quality attributes such as required refinement, design, and model links/relationships to other requirements. These attributes and their specific values define measurable completeness levels of requirements or specification documents and enable quality and progress control in RE. This also supports the search for vague or volatile requirements and evaluation; e.g., the rules described for the goal-oriented classification concept can be used for analyzing the completeness and consistency of specified requirements.

  Return of investment (ROI) and evaluation of *alternative concepts*

  The structured system modeling together with the goal-oriented tracing of the RE artifact model enables meaningful feasibility studies and effort/cost estimation. A structured model is the basis for systematic cost estimation, either with

CoCoMo or other methods. Such a model is also useful for estimating the benefits of a system (e.g., savings and efficiency gains, market position improvement, sales increase). In effect, it supports sophisticated consolidation and conclusion about the ROI of requirements or alternative system concepts and their implementation.

- *Risk analysis*

  The validation and verification concepts of REM are also a basis for a systematic risk analysis. Incomplete, unstable, vague, conflicting, or poorly consolidated requirements indicate possible project risks. Based on the structure of the RE artifact model these risks can be classified, analyzed, and evaluated. Also, systematic mitigation strategies can be developed. For a formal risk analysis, the impact (measured in time lost or in costs) and probability of the occurrence for every risk factor can be estimated.

- *Requirements prioritizing*

  The analysis views systematically support comprehensive analysis and consolidation of user requirements and system concepts. Together with additional techniques for customer/market-oriented product definition, such as Kano Modeling [Ka+84], the REM approach supports the implementation prioritization of requirements.

Method-specific analysis views

Method-specific or *advanced analysis views* use additional modeling approaches to develop and analyze requirements. These modeling approaches can be characterized as advanced or application-specific aspects. Some of them are current research findings or useful only for specific application domains. Their use is recommended for specific systems and project characteristics (see the tailoring concept in Chapter 3). The application of these modeling approaches requires experienced/skilled people and the support of proven techniques/tools. As shown below, these approaches are built up with the basic structuring rules of the RE artifact model and then extended by specific aspects to analyze requirements.

- *Refinement/Modeling*

  The major task of RE is the analysis, development and consolidation of requirements. The initially given goals, functional and non-functional requirements must be captured, evaluated, refined, structured, and appropriately represented by system models. Promising approaches to support specific aspects of this refinement are modeling concepts such as *feature modeling* and *quality modeling.*

  Feature modeling refines and structures customer-oriented requirements (features) with the help of feature trees and their modeling rules. Figure 14 shows an example of a FODA feature tree [KC+90]. It specifies possible instances and combinations of system features. This technique models the characteristics of a specific application domain or a group of systems that belong together within a product-line or platform. A main goal of this kind of feature modeling is the definition of possible sellable feature combinations that define allowable system configurations/versions in terms of variation points and feature composition rules. Other goals of feature modeling are the reuse of these domain specific models within a product-line development and for determining the scope of the current project.

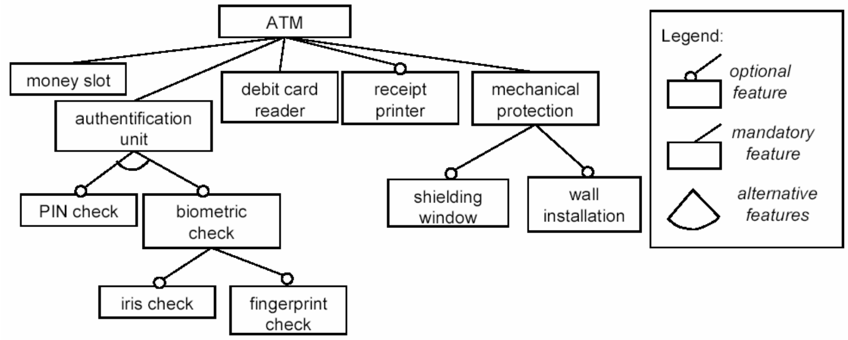
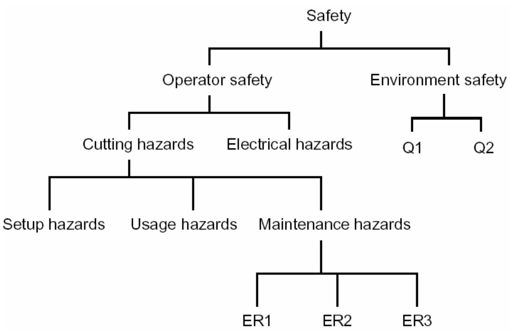


Figure 14: FODA Feature Model Example [KC+90]

The goal of *quality modeling* is to refine abstract quality requirements such as safety or usability and derive corresponding concrete functional requirements and quality constraints of the system. The goal is to derive a comprehensive quality profile for the SuC. Figure 15 shows the refinement of the *safety concern* for a paper guillotine [SS98]. The resulting quality tree works as a checklist for further requirements definition within the same application domain. Therefore, it defines a domain specific quality model.



Req	Name	Description
ER1	Accidental startup	The control system must disable the guillotine so that it cannot be started accidentally during the maintenance process.
ER2	Blade installation	The control system must include a visual indicator to show that both the principal and backup blade fixing mechanisms are properly secured. The guillotine must be disabled until both principal and backup blade fixing mechanisms are secured.
ER3	Blade removal	The control system must ensure that the blade shield is in place before the blade lock may be opened.

Figure 15: Safety-Concern Refinement of a Paper Guillotine [SS98].

Another purpose of quality modeling is the evaluation or verification of already designed system architectures regarding the required quality attributes or concerns. Figure 16 shows an example of the Utility Tree of the ATAM Method [KK+00]. It is used to refine domain specific quality requirements and derive testable application scenarios for the designed system.

The primary purpose of these methods is the refinement of “high-level” business requirements to structured and measurable system requirements by the analysis and description of detailed functional system specifications as they are provided by REM.

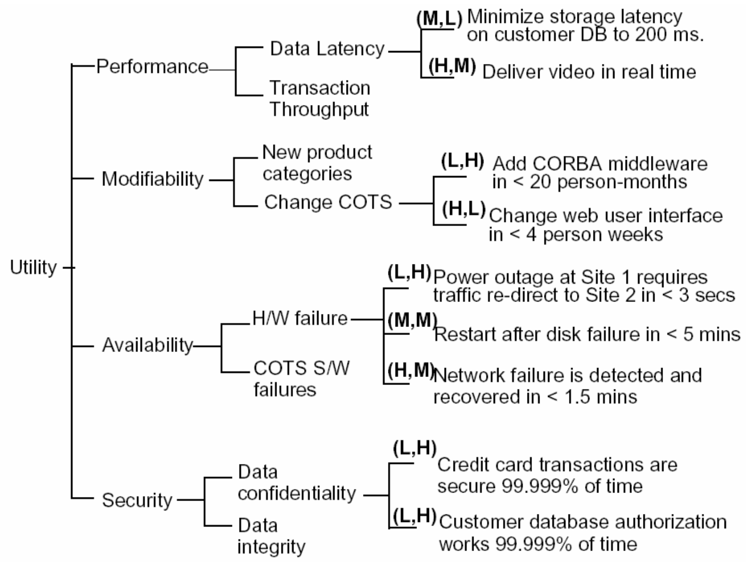


Figure 16: Utility Tree – Example of the SEI ATAM Approach [KK+00].

- *Prototyping, validation and verification based on functional system modeling*

  The functional system views with their consistency relations (see section 4.1 above) can be informally used by drawing and discussing the system requirements with the help of sketches or semi-formal description techniques such as use cases described in UML [UML04]. This functional system modeling allows the specification of executable models, and therefore supports prototyping and direct feedback from potential users or design engineers of the system.

  Advanced modeling support is given by more formal RE and specification approaches and tool support that seamlessly integrate requirements management and formal system modeling. Because of their mathematically defined description techniques and consistency rules these approaches support comprehensive verification and validation of requirements. As realized by the AutoRAID approach [AR04, AutoFocus, AF2], formal system modeling serves as a prototype of sophisticated evaluation of:

  - specified system behavior,
  - hazard analysis,
  - evaluation of alternative system concepts, and
  - technological risk.

Additional and Adapted RE Artifacts

As shown by the refinement approaches, the application of advanced or specific analysis methods depends on the characteristics of the actual project or domain. For example, feature modeling is reasonable in cases of product-line development and hazard analysis is required for safety critical domains. These specific analysis activities result in additional or diverging RE artifacts to the elements of the core RE artifact model. Therefore, it is necessary to tailor and instantiate REM to specific domains and project needs (see also Chapter 3).

Chapter 5 Summary and Outlook

Defining appropriate system requirements is a key enabler to achieve business success. Requirements engineering is one of the keys to mastering the complexity of today's software intensive systems. Requirements engineering excellence is therefore a must capability for enterprise businesses to implement in the medium- to the long-term.

RE is a fairly young and interdisciplinary discipline involving many organizational stakeholders. Currently, RE is more an art and sometimes a handicraft rather than an engineering discipline. As a result, RE excellence today often depends on a small number of skilled personal.

For achieving RE excellence, there is no convenient yardstick available. There are many ingredients and success factors for RE. Among others, a framework is needed that can be used as the basis of an incremental strategy to achieve RE excellence.

One of the leading research institutions, Technische Universität München (TUM), is partnering with, Siemens Corporate Research Princeton to develop such a RE reference approach called the Requirements Engineering Reference Model (REM).

REM defines the core set of RE work products and their dependencies. REM helps guide the establishment and maintenance of process- and project-specific RE methodologies, processes, and tools. REM consists of:

- the definition of an artifact model (a set of work products resulting from requirements engineering activities and their dependencies), and
- a tailoring approach for both the artifact model and process, the methods and activities that are applied, and the way the artifacts are represented and structured as documents.

Within Siemens it is the goal to use REM as a yardstick for measuring, comparing, evaluating and improving RE practice, methods and tools starting with the project experience of Siemens Corporate Resarch (SCR) resulting from their consulting practice, but also the research projects of TUM within the automotive industry. Retrospective analysis of projects will lead to a collection of RE best practices

Tailoring is the means to achieve a standardized approach to RE, with all the advantages of a unified methodology, and to avoid at the same time the "one-size-fits-all" pitfall. Tailoring is done to address the needs of specific application domains and project characteristics.

REM provides an artifact oriented RE standard process with a tailoring approach, in order to define specific work products, document structures and RE process definitions that are appropriate to the specific project and domain needs. This includes the assignment of adequate methods and description techniques to support communication and reasonable formality. Within REM, tailoring is done in the following dimensions:

- trimming the product model
- structuring the work products into a RE document set
- selecting methods for defining and representing work products
- deciding on the process in terms of the document structure.

Currently, RE projects are mainly done based on informal methods for generating the work products. The TUM projects used formal methods and model-driven development, which allow for establishing formal relations between the work products. This will give us useful hints how to use formal methods and under which circumstances they pay off.

In practice, we find classical waterfall process models, iterative processes, and also agile processes. REM is process agnostic so all these different process concepts can be described within the framework, and they can be compared to analyze the potential benefits and shortfalls of the chosen processes.

REM is only a step in the direction of a comprehensive RE approach. One important goal of future work is to understand how the RE methods can complement each other, and where they should or should not be applied to address specific RE issues depending on the needs of the specific domain and other project characteristics. Another goal is to understand better how RE fits into the overall product lifecycle process, and how development organizations and their software projects are structured to effectively facilitate the use of RE methods. In particular, REM assures that all relevant issues of RE are considered when designing individual RE processes.

REM supports the development of software intensive systems (from definition to maintenance) including the upstream activities to explore alternate system concepts. REM can be used for any type of project such as embedded systems within technical devices and software systems that are embedded in organizational or technical processes.

REM gives methodological support to carry out the tailoring. It is briefly outlined in this document. In the future, a number of customizations of REM, e.g. embedded systems, IT-systems, are planned.

Establishing a workable REM approach will constitute a significant contribution to the field of RE research. Furthermore, it will enable enterprises business to piggy-back their improvements in RE on proven best practices and models.

REM is just one step, with others to follow. We plan and foresee the following research and development activities:

- *instances of REM for specific domains and selected process models*
- *methods for describing the various REM work products and improving those methods*

- *providing tool support for REM (beyond prototypes like AutoRAID)*
- *incorporating REM into overall life cycle models (such as V-Modell XT).*

There is still plenty of work to do in fundamental and applied research as well as defining practical methods and improving the RE state-of-practice.

Bibliography and References

[Aka90] Akao, Y. (Ed.): Quality Function Deployment: Integrating Customer Requirements into Product Design. Translated by Glenn Mazur. Cambridge, MA, Productivity Press, 1990.

[AG90] Ashworth, C., Goodland, M.: SSADM A Practical Approach. McGraw-Hill, 1990.

[AutoFocus] Website of AutoFocus with documentation, screenshots, tutorials and download: http://www4.in.tum.de/~af2/index.shtml

[AF2] Wild, D.: AutoFocus 2 – Das Bilderbuch. Technische Universität München, Technical Report: TUM-I0610, May 2006.

[AR04] Website of AutoRAID project, with documentation, screenshots and downloads: http://wwwbroy.in.tum.de/~autoraid/. Current downloads of AutoRAID/AutoFocus available on the AutoFocus-homepage: [AutoFocus].

[BK+01] Bass, L., Klein, M., Moreno, G.: Applicability of General Scenarios to the Architecture Tradeoff Analysis Method. Technical Report. CMU/SEI-2001-TR-014. ESC-TR-2001-014.

[Ber04] Berenbach, B.: The Evaluation of Large, Complex UML Analysis and Design Models. Proceedings of the 26th Int. Conf. on Software Engineering, ICSE 2004, Edinburgh.

[BS02] Bittner, K., Spence, I.: Use Case Modeling. Addison Wesley, 2002.

[Bos00] Bosch, J.: Design & Use of System Architecture – Adopting and Evolving a Product-line Approach. Addison-Wesley Professional, 2000.

[Bos04] Bosch, J.: On the Development of Software Product Family Components. Proceedings of the Third Conference Software Product Line Conference (SPLC 2004), Springer Verlag LNCS 3154, pp. 146-164, September 2004.

[BP+99] Bourdeau, E., Lugagne, P., Roques, P.: Hierarchical Context Diagrams with UML: An Experience Report on Satellite Ground System Analysis. Springer, Lecture Notes in Computer Science, 1999.

[BD+06] Broy, M., Deissenboeck, F., Pizka, M.: Demystifying Maintainability. WoSQ '06: Proceedings of the 4th Workshop on Software Quality, 2006.

[BS01] Broy, M. Stoelen, K.: Specification and Development of Interactive Systems. Springer, 2001.

[C++Style] Henricson, M., Nyquist, E.: Programming in C++: Rules and Recommendations. Technical Report, Telecommunication Systems Laboratories, 1992.

[Car95] Carroll, J.: Scenario-Based Design: Envisioning Work and Technology in System Development. Wiley & Sons, 1995.

[CL99] Constantine, L., Lockwood, L.: Software for Use: A Practical Guide to the Models and Methods of Usage-Centered Design. ACM Press, 1999.

[CN+00] Chung, L., Nixon, B., Yu , E., Mylopoulos, J.: Non-Functional Requirements in Software Engineering. Kluwer Academic Publishing, 2000.

[CO+06] Creighton, O., Ott, M., Bruegge, B.: Software Cinema–Video-based Requirements Engineering. 14th IEEE International Requirements Engineering Conference, Minneapolis/St. Paul, 2006.

[DeM79] DeMarco, T.: Structured Analysis and System Specification. Prentice-Hall, 1979.

[DP+03] Denger, C., Paech, B., Benz, S.: Guidelines – Creating UCs for Embedded Systems. IESE-Report No. 078.03/E, 2003.

[DG96] Dinkhoff, G, Gruhn, V.: Entwicklung Workflow-Management-geeigneter Software-Systeme. In G. Vossen, J. Becker (Hrg.): Geschäftsprozeßmodellierung und Workflow-Management – Modelle, Methoden, Werkzeuge, pp. 405-421. International Thomson Publishing, Bonn, Albany, 1996.

[DK+05] Doerr, J., Kerkow, D., Koenig, T., Olsson, T., Suzuki, T.: Non-Functional Requirements in Industry – Three Case Studies Adopting the ASPIRE NFR Method. IESE-Report No. 025.05/E, 2005.

[EN96] Easterbrook S., Nuseibeh B.: Using ViewPoints for Inconsistency Management. Software Engineering Journal, 11, 1, BCS/IEE Press, pp. 31-43, 1996.

[Gei05] Geisberger, E.: Requirements Engineering Eingebetteter Systeme - ein interdisziplinärer Modellierungsansatz. Dissertation, Technische Universität München, 2005. Shaker Verlag, ISBN: 3-8322-4619-3, www.shaker-online.com/.

[GG+06] Geisberger, E., Grünbauer, J., Schätz, B.: Interdisciplinary Requirements Analysis Using the Model-based RM Tool AutoRAID. Proceedings of the Automotive RE Workshop. IEEE International RE Conference 2006, Minneapolis, 2006.

[GNUStyle] Stallmann, R.: GNU Coding Standards. http://www.gnu.org/prep/standards_toc.html , 2001.

[HK+04] Halmans, G.; Kamsties, E.; Pohl, K.; Reis, S.; Reuys, A.: Seamless Transition from Requirements to Test Cases: How to Test a Software

Product Line? In: Proceedings of the Conference on Software Testing, ICSTEST-E (Bilbao, Spain, November 2004). SQS, Düsseldorf 2004.

[HB00] Heitmeyer, H., Bharadwaj, R.: Rigorous Requirements for Safety-Critical Systems: Fundamentals and Applications of the SCR Method. Tutorial on ETAPS 2000 (European Joint Conferences on Theory and Practice of Software), Berlin, 2000.

[HN+05] Hofmeister, C., Nord, R. L., Soni, D.: Global Analysis: Moving from Software Requirements Specification to Structural Views of the Software Architecture. IEE Proceedings Software, Vol. 152/ 4, pp. 187-197, 2005.

[HP05] Herrmann, A., Paech, B.: Software Quality by Misuse Analysis. Technical Report SWEHD-TR-2005-01, 2005.

[HP+97] Hsia, P., Kung, D., Sell, C.: Software Requirements and Acceptance Testing. In Mead, N.: Annals of Software Engineering, ed. 3, pp. 291-317, 1997.

[IBMREF] Adams, J., Koushik, S.: Patterns for E-Business: A Strategy for Reuse. MC Press, 2001.

[IEEE90] IEEE-Standard: Std. 610.12-1990.

[Jac95] Jackson, M.: Software Requirements & Specification: A Lexicon of Prac-tice, Principles and Prejudices. Addison-Wesley, 1995.

[JavaStyle] Sun Microsystems: Java Look and Feel - Code Conventions for the Java Programming Language. Technical Report, 1999. http://java.sun.com/docs/codeconv.

[Jon94] Jones, C.: Assessment and Control of Software Risks. Yourdon Press Computing Series, 1994.

[Jon98] Jones, C.: Estimating Software Costs. McGraw-Hill, 1998.

[J2EE] J2EE: http://java.sun.com/javaee/index.jsp

[Ka+84] Kano, N. et al. Attractive Quality and Must-be Quality, in: Hinshitsu: The Journal of the Japanese Society for Quality Control, 4/1984, pp. 39-48.

[KC+90] Kang, K., Cohen, S., Hess, J., Novewak, W., Peterson, A.: Feature – Oriented Domain Analysis (FODA) Feasibility Study. Technical Report CMU/SEI-90-TR-021, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, 1990.

[KK+00] Kazman, R., Klein, M., Clements, P.: ATAM: Method for Archtecture Evaluation. Technical Report, CMU/SEI-2000-TR-004.

[Kof05] Kof, L.: Text Analysis for Requirements Engineering. PhD thesis, Technische Universität München, 2005.

[KC+02] Konrad, S., Campbell, L. A., Cheng, B.: Adding formal specifications to requirements patterns. In Proceedings of the Requirements for High Assurance Systems Workshop (RHAS02) as part of the IEEE Joint International Conference on Requirements Engineering (RE02), Essen, Germany, September 2002.

[KC02] Konrad, S., and Cheng, B.: Requirements Patterns for Embedded Systems. Proceedings of the Tenth IEEE Joint International Requirements Engineering Conference (RE'02), Essen, September 2002.

[KS95] Kotonya, G., Sommerville, I.: Requirements Engineering With Viewpoints, Technical Report, Lancester University, 1995.

[KS98] Kotonya, G., Sommerville, I.: Requirements Engineering: Processes and Techniques, John Wiley & Sons, 1998.

[Kru95] Kruchten, P.: The 4+1 View Model of Architecture. EEE Computer Society Press, 1995.

[LD+98] Lamsweerde, A., Darimont, R., Letier, E.: Managing Conflicts in Goal-Driven Requirements Engineering. IEEE Trans. Software Eng. 24(11), pp. 908-926, 1998.

[LF91] Leite, L., Freeman, P.: Requirements Validation Through Viewpoint Resolution, In: IEEE Transactions on Software Engineering, 1991.

[LH+94] Leveson, N., Heimdahl, M., Hildreth, H., Reese, R.: Requirements Specification for Process-control Systems. IEEE Transaction on Software Engineering, pp. 684-706, 1994.

[LR+97] Leveson, N., Reese, R., Koga, Sh., Pinnel, D., Sandys, S.: Analyzing Software Specifications for Mode Confusion Potential. In Proceedings of the Workshop on Human Error and System Development, 1997.

[LK95] Loukopoulos, P., Karakostas, V.: Systems Requirements Engineering. McGraw-Hill, 1995.

[MOF06] OMG Meta Object Facility homepage: http://www.omg.org/mof/.

[MP84] McMenamin S.M., Palmer J.F.: Essential Systems Analysis. Yourdon Press. Prentice Hall, 1984.

[MG+60] Miller, G., Galanter, E., Pribram, K.: Plans and the Structure of Behavior. New York: Holt, Rinehart & Winston, 1960.

[MT+06] Miller, S., Tribble, A., Whalen, M., Heimdahl, M.: Providing the Shalls. International Journal on Software Tools for Technology Transfer (STTT), Feb 2006.

[MSF02] Microsoft Solution Framework (MSF) Team Model. White Paper. 2002. Homepage: http://www.microsoft.com/msf

[NK+94] Nuseibeh, B., Kramer, J., Finkelstein, A.: A Framework for Expressing the Relationships between Multiple Views in Requirements Specifications. IEEE Trans. On Software Engineering. 20(10), pp. 760-773, 1994.

[PD+02] Paech, B., Detroit, A. H., Kerkow, D., von Knethen, A.: Functional Requirements, Non-functional Requirements, and Architecture Should Not be Separated - A Position Paper. REFSQ' 2002, Essen, Germany, 2002.

[PK+03] Paech, B., Von Knethen, A., Doerr, j., Bayer, J., Kerkow, D., Kolb, R., Trendowicz, A., Punter, T., and Dutoit, A.: An Experience-Based Approach for Integrating Architecture and Requirements Engineering, ICSE'03 Workshop "From Software Requirements to Architectures", 2003.

[Par92] Parnas, D.L.: Tabular Representation of Relations. CRL Report 260, Communications Research Laboratory, McMaster University, October 1992 .

[PM95] Parnas, D., Madey, J.: Functional Documentation for Computer Systems. Science of Computer Programming 25, pp. 41-61, Oktober 1995.

[Pat99] Paterno, F.: Model-Based Design and Evaluation of Interactive Applications. Springer, 1999.

[PB+01] Pohl, K.; Brandenburg, M.; Gülich, A.: Integrating Requirement and Achitecture Information – A Scenario and Meta-Model Based Approach. In: Proc. of REFSQ 2001 Workshop. Interlaken, Schweiz, 2001.

[PH97] Pohl, K.; Haumer, P.: Modelling Contextual Information about Scenarios. In: Proc. of the 3rd Intl. Workshop on Requirements Engineering – Foundation of Software Quality (REFSQ `97). University Press Namur, Barcelona, Spain, 1997.

[PS05] Pohl, K.; Sikora, S.: Requirements Engineering für eingebettete Software. In: P. Liggesmeyer, D. Rombach (Hrsg.): Software Engineering eingebetteter Systeme – Grundlagen - Methodik - Anwendungen. Elsevier, Heidelberg 2005.

[QUASAR] http://www.first.fraunhofer.de/quasar.

[QuasarREF] Siedersleben, J.: Moderne Software-Architektur. Umsichtig planen, robust bauen mit Quasar. Dpunkt Verlag, 2004.

[RR99] Robertson, S., Robertson, J.: Mastering the Requirements Process. Addison-Wessley, 1999.

[RB+98] Rolland, C., Ben Achour, C., Cauvet, C., Ralyte, J., Sutcliffe, A., Maiden, N., Jarke, M., Haumer, P., Pohl, K., Dubois, E., and Heymas, P.: A Proposal for Scenario Classification Framework. Requirements Engineering Journal, Vol. 3 No. 1, 1998.

[Rup02] Rupp, C: Requirements Engineering und Management, Hanser Verlag, 2002.

[Sch04] Schätz, B.: Mastering the Complexity of Embedded Systems - The AutoFocus Approach. In: Fabrice Kordon, F., Lemoine, M.: Formal Techniques for Embedded Distributed Systems: From Requirements to Detailed Design. Kluwer, 2004.

[Sch02] Scheer, A.: ARIS – Vom Geschäftsprozess zum Anwendungssystem. Springer-Verlag, Berlin, 2002.

[SM+06] Seybold, C., Meier, S., Glinz, M.: Scenario-Driven Modeling and Validation of Requirements Models. 5th ICSE International Workshop on Scenarios and State Machines*: Models, Algorithms and Tools*, Shanghai, May 2006.

[S PLM] Siemens Product Lifecycle Management (PLM) homepage: https://cio.siemens.de/index.jsp?sdc_p=cfi1000000001l0m1o2000001232120ps8u50z&sdc_bcpath=2000001047932.s_8%2C2000001234946.s_8%2C%3B2000001232120.s_8%2C&sdc_sid=28656265239&

[Som04] Sommerville, I.: Software Engineering. 7th Edition. Addison Wesley, 2004.

[SS98] Sommerville, I., Sawyer, I: "Viewpoints for Requirements Elicitation: a Practical Approach", Proc. IEEE Int. Conf. on Requirements Engineering, Colorado Springs. IEEE Press, P. 1998.

[SM+05] Song, X., Matos, G., Hwong, B., Rudorfer, A., Nelson, Ch.: S-RaP: A Concurrent Prototyping Process for Refining Workflow-Oriented Requirements. In Proceedings of IEEE Int. Requirements Engineering Conference 05, Paris, 2005.

[Sul86] Sullivan, L.: Quality Function Deployment. Quality Progress, June 1986, pp 39-50, 1986.

[SUNREF] http://www.sun.com/service/refarch/

[Sut02] Sutcliffe, A.: User-centered Requirements Engineering – Theory and Practice. Springer, 2002.

[SySML] Open source System Modeling Language (SySML) homepage: http://www.sysml.org/

[TROPOS] www.troposproject.org/.

[TTCNTest] European Telecommunications Standards Institute (ETSI). ETSI ES 201 873: TTCN-3, Edition 3.1.1. http://www.ttcn-3.org/Specifications.htm, June 2005.

[UML] http://www.uml.org/.

[UML2Test] Object Managment Group, Inc. UML Testing Profile, Version 1.0. http://www.omg.org/technology/documents/formal/test_profile.htm, July 2005.

[VM-XT] V-Modell XT. http://www.v-modell-xt.de/.

[WP+98] Weidenhaupt, K.; Pohl, K.; Jarke, M.; Haumer, P.: Scenario Usage in System Development – A Report on Current Practice. IEEE Software, Vol. 15, No. 2, p. 34-45, 1998.

[Wie99] Wiegers, K., E.: Software Requirements. Microsoft Press. 1999.

[.NET] .Net: http://www.microsoft.com/net/default.mspx

Apendix A History, Background and Related Work

In the following paragraphs it is shortly outlined what is the basis of REM and how it relates to specific RE approaches, standards and RE activities within Siemens.

A.1 Approach to Developing REM

The approach to develop REM is built on a scientifically sound basis and experiences in a broad range of Siemens RE projects:

- **Identify and analyze "best-practices" of RE in different industrial sectors.** By this, we can effectively identify what features of a RE method is particularly useful for addressing certain requirement analysis needs. REM is fully based upon methods proven and successfully applied in practice.

- **Identify major RE components and methods.** As the major RE methods those have been selected that have a solid bases and have been practiced successfully. Thus, identifying RE methods and integrating such into REM will ensure the REM completely enough to capturing all major RE method features.

- **Develop and validate REM to examine if it encompasses all major RE components.** For a given RE feature, we will check if REM models it. With such a validation, we ensure the completeness of REM.

The basic philosophy of REM as a tailorable standard artifact model is very much along the lines of the V-Model XT (see [VM-XT]). The core content of REM stems from research work at the Technische Universität München, in a number of projects and in particular, the Ph. D. Thesis by Eva Geisberger (see [Gei05]) as well as the body of knowledge in SCR, in particular, Brian Berenbach.

A.2 Background of REM and Relationship to State of the Art

A.2.1 Sources of Approach

The REM approach draws its analysis and modeling concepts from three areas:

- Research projects of the Technische Universität München (TUM) in the field of system modeling, its theory (Focus [BS01]) and its tooling AutoFocus [Sch04], [AutoFocus]

- TUM approaches in the field of process definition (CMMI assessments, V-Model XT [VM-XT]) and Requirements Engineering, its methodical concept, REFocus, and its model-based requirements management support AutoRAID/AutoFocus [AutoFocus].

- Intense cooperation with industry, particularly Siemens companies, where the approaches have been tested and further developed. Specific input is adapted from the Siemens SCR projects Design Advisor [Ber04] and S-RaP [SM+05].

REM is a typical best of breed approach taking into account many results and insights produced by theory and practice.

A.2.2 Relation to Specific Approaches

There are specific RE approaches for the following sub-areas of RE:

- eliciting and modeling requirements (e. g. Kano-modeling [Ka+84], Global Analysis approach [HN+05], linguistic methods [Rup02, Kof05] , Quality Function Deployment (QFD) [Sul86, Aka90], Video-based RE [CO+06], Use-Case and scenario modeling [PH97, PB+91, HK+04], [RB+98], SEGOS RE [PS05], [HP05], QUASAR [QUASAR, DP+03]), Scenario and user task modeling: [Car95], [Sut02], [Som04], [Pat09], Viewpoint modeling ([KS95, SS98], [LF91]), domain analysis [Jac95], functional system modeling and decision tables [PM95, Par92], [LH94, LR97])

- goal and quality refinement (e. g. VORD/PREView [KS95, KS98, SS98], the KAOS-approach [LD+98], the TROPOS-approach [TROPOS], ATAM-Method [KK+00]), ASPIRE-method [DK+05], MOQARE-method [HP05])

- multiple views and inconsistency management (e.g. [LF91], [NK+94], [EN96], [KS95, SS98], [Ber04] and further methods based on (semi-) formal requirements and system modeling)

- formal system modeling and specification (e.g. [Par 92, PM95], SCR [HB00], Focus [BS01, Sch04], KAOS [LD+98], SpecTRM [LH+94, LR+97]), [KC02, KC+02])

- integrated RE and system modeling (e.g. scenario-based approaches [PB+01], [PD+02, PD+03], [SM06], based on formal system modeling [KC02, KC+02], [MT+06]

- providing content structures and templates of specification documents (e.g. VOLERE template [RR99] QUASAR document templates [QUASAR])

- product-line RE (e. g. Feature-oriented Domain Analysis (FODA) [KC+90], scenario-based derivation of test cases [HK+04], ScenTED [HK+04]), architecture-design [Bos00, Bos02].

They focus on specific application domains or problems in RE.

In contrast, REM constitutes an RE framework that covers the core elements of RE and supports the integration of specific RE methods and best practices. Their effective use is secured by the tailoring concept of REM. Individual approaches to core areas of RE are incorporated.

A.2.2 Relation to Standards and RE Efforts within Siemens

Relationship of REM to Capability Maturity Model Integrated (CMMI)

CMMI provides a standard framework for defining system development processes. Within Siemens, the Operating Companies are aiming at achieving CMMI level-3 by end of 2007. REM builds upon the core concepts of CMMI and reuses a large set of its terminology.

REM is compliant to CMMI. More particularly, REM complements what is required in CMMI for RE to a standard RE model.

Relationship of REM to Product Life-Cycle Management (PLM)

PLM [S PLM] stands for the process-driven view on a product, which encompasses the life-cycle. It is a Siemens corporate standard for Operating Companies to be compliant in the organizational set-ups. The expected benefit of using PLM is to achieve ease of doing business between the Operating Companies as well as work together to deliver high business value to their customers. REM provides a concrete way to define and manage product requirements. Further, REM is compliant to the goals and supports the PLM.

Apendix B REM Philosophy

REM is built on two fundamental presumptions:

- RE is an iterative process of problem statement and solving
- A common understanding about systems command how to describe them: a common system concept

The following paragraphs shortly outline these basic concepts.

B.1 RE as an Iterative Process of Problem Statement and Solving

RE corresponds to the iterative tasks of problem solving: To construct and precisely specify a problem-focused system specification the following interrelated tasks have to be carried out:

- Understand, analyze the problem and accurately define the user, customer and business goals of the system under consideration (SuC). This includes capturing and consolidating the product, marketing and business requirements and drawing up a vision of the system concept.
- Analyze, refine, consolidate, and specify the requirements and conditions of the SuC. This includes prioritizing and consolidation of conflicting goals/requirements of the stakeholder as well as understanding and decision making about the constraints and context of the SuC.
- Based on the developed understanding, find, construct and decide upon the system concepts that best meet defined goals, requirements and constraints. Accurately specify the detailed system requirements.

Understanding the problem is prerequisite for constructing a solution concept. Therefore, these tasks are closely connected and performed iteratively.

One has to keep in mind that RE implies repeatedly making choices and taking crucial decisions about

- which functionality in which quality is required,
- the detailed way in which the required functionality is going to be realized in a system concept and to what degree that satisfies the quality goals and context constraints of the SuC.

This decision making process is crucial for the properties, the acceptance and the cost of the system and thus the success of a project.

Importance of Interdisciplinary Communication

The stakeholders in the RE process focus on different aspects of product requirements as well as use different ways to view them. For importance, people from marketing or product management view a system from the market or user perspective while engineering has its mind set on figuring out how to technically implement product requirements and construct a system concept. Interdisciplinary communication and consolidation of different views on the problem and solution concepts is a prerequisite for making the right decisions in system definition. A systematic RE aims at achieving measurable consistency constraints that support the diagnosis of inconsistencies, vagueness or conflicts as a necessary starting point for validation and consolidation of requirements.

REM provides a core set of RE specification work products and dependencies. As requirements engineering is a problem solving task, in addition to the specification concepts of systems and system requirements, REM includes the specification concepts of goals and analysis/refinement models of high level requirements and general conditions. Because it is a goal-oriented approach, specified requirements, system concepts and constraints must be justified by business goals and the result of explicit prioritizing and decision making of conflicting requirements and alternative system concepts. The core set of RE specification work products are structured according to these dependencies thus they form a reference model for interdisciplinary consolidation of requirements and decision making in RE.

REM supports systematically control of the ongoing decision process by the concept of an iterative feedback loop. It evaluates the quality of the so far worked out specifications and provides risk analysis and prioritizing of requirements as prerequisite to appropriate decision making.

B.2 System Concept

When applying REM it is always assumed that there is a system under consideration (SuC) for which the requirements are worked out. The system can be part of a product. Typically for the SuC there exists an environment under consideration (EuC). Generally, the SuC itself is structured hierarchically (into sub-systems). In RE, one of the main goals is the specification of the functional behavior of the SuC in relation to the EuC. These interaction specifications are called the functional requirements. In addition, further non-functional requirements (NFR) are captured that cover quality aspects and constraints. For the EuC all assumptions are documented that are relevant for the SuC. All detailed requirements and constraints are derived from high-level requirements and the business goals. These goals provide the guidelines for capturing detailed requirements and constraints.

In RE, the identification of the SuC and EuC are major steps. The system boundaries in the SuC have to be worked out explicitly and well documented. For the EuC it has to be identified, determined and documented which properties, aspects, and issues of the system context are relevant for the SuC.

To understand and determine this functional integration of the SuC into the EuC, REM identifies, considers and structures the relevant part of the environment again as system with sub-systems – the affected domain. For example, if the SuC is a locking system of a car, then the SuC will use or will be used by other systems of the overall car environment, e.g. the steering system, sensor systems or safety systems etc. Also, this is needed to systematically identify and determine the various levels of logical and sensitive man-machine interfaces of the SuC and the car (overall system).

Apendix C TUM Prototypical Instance of REM

This appendix outlines the TUM-specific methodical instance of REM by the data model structure of AutoRAID/AutoFocus. It is a prototypic tool for integrated RE and system specification. It is based on a mathematically well-founded system model [BS01] and the result of comprehensive research in model-based development for embedded systems at the TUM.

Figure C-1 illustrates the user interface of AutoRAID by showing the *Analysis*-tree of the project *RevMeter* and a generic data structure sheet of the requirement *Display RPM* (*Type, Title, Description, Patron, Status, Priority, Rational*, Lists of *Sub-* and *Super-Requirements*). Depending on the *Type* of the requirement this structure is extended by specific descriptions (specifying use cases, scenarios steps or relations to functional system model elements, see data models in figure C-2 and C-4).

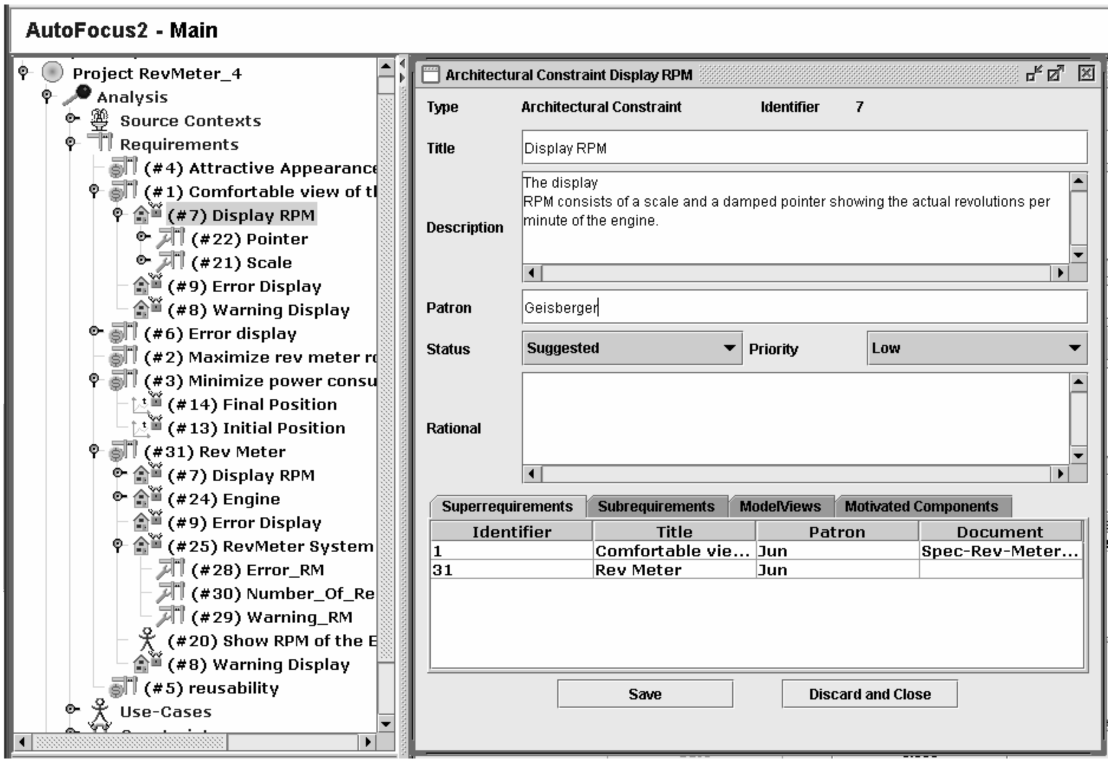


Figure C-1: User Interface of AutoRAID

A detailed description of AutoRAID/AutoFocus and its intended use can be found in [GG+06] and [Sch04].

Figure C-2 and C-4 provide the formal data model for the integrated requirements management and system modeling concept of REM. It describes the methodical basis of the prototype AutoRAID/AutoFocus and defines dependencies between the RE and system modeling artifacts. Thus it specifies tool support for measurable progress and quality in RE and enables appropriate project steering and control.

Figure C-3 illustrates the corresponding functional analysis and system specification concept of analyzing system use scenarios and deriving required system *services* with the help of

- the complementary system modeling views of *Process, Interaction, Behavior, Structure (*Environment Model, System Boundaries), *Data* modeling and their

- model relations and dependencies (shown in Figure C-4).

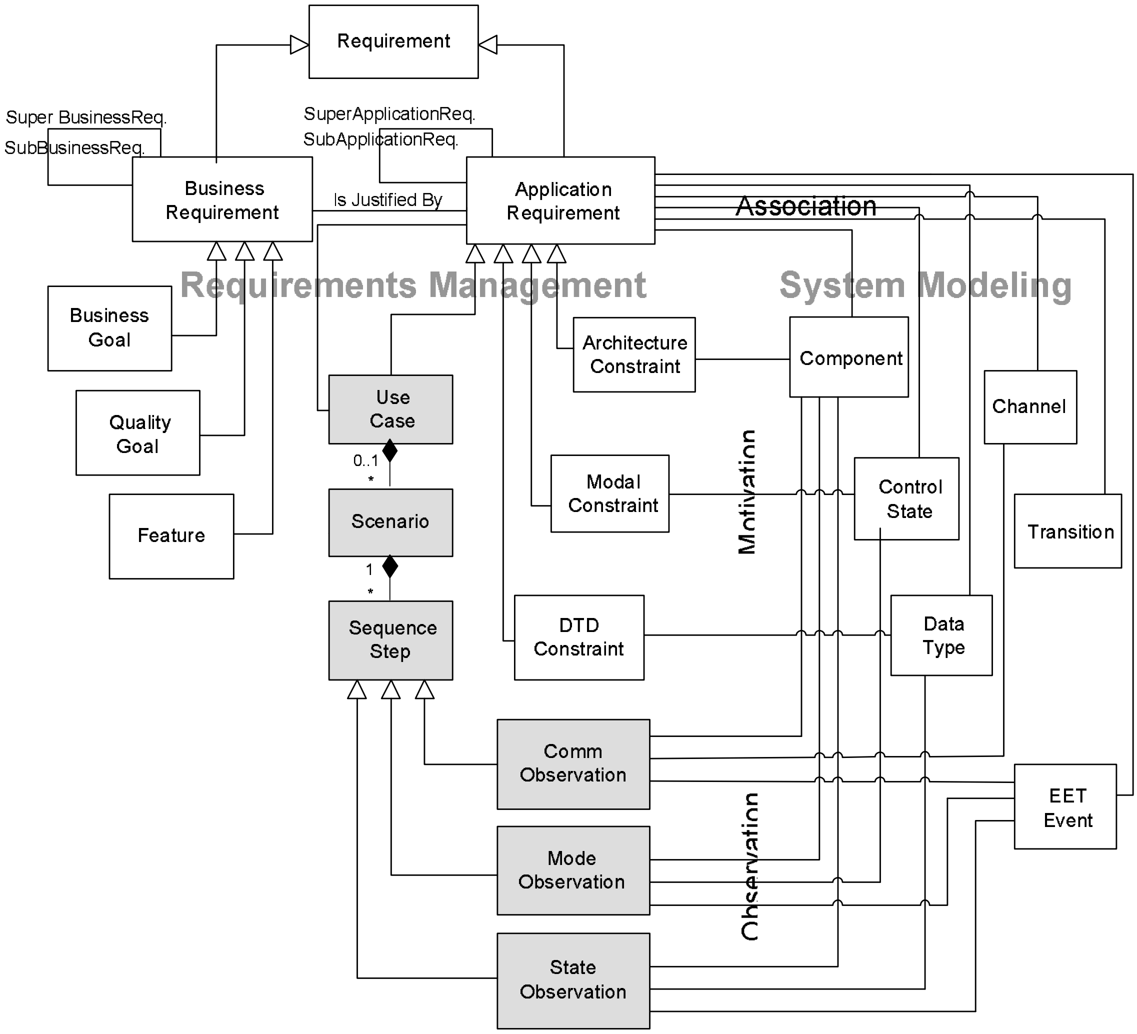


Figure C-2: Requirements Management (RM) data model of REM using the System Modeling elements of AutoFocus [AutoFocus, AF04]

According to the RE Artifact Model structure described in Chapter 4, the overall data model of AutoRAID/AutoFocus in Figure C-2 is separated into two parts: Requirements Management (on the left) and System Modeling (on the right):

- the left part specifies the goal-oriented refinement and classification of requirements with the help of use case/scenario modeling and the functional classification of requirements (*Use Case, Constraints*), and

- the right part shows the construction and detailed specification of the functional solution concept by this functional classification of requirements and the corresponding *Motivation* and *Association* relation between requirements and system modeling elements.

Major technique for deriving detailed system services (*Use Case*), evaluating constructed system model and further refining/revisioning of requirements and system specifications is the process and system behavior analysis by observations (*Observation* relations) of single *scenario steps* (*Communication, Mode, State Observation*).

Goal-oriented Refinement of Requirements

According to the goal-oriented refinement of REM*, Requirements* are classified as *BusinessRequirements* or *ApplicationRequirements*, and *ApplicationRequirements* must be *justified by BusinessRequirements.* Both kinds of requirements can be refined and structured by *SuperRequirements*- and *SubRequirements*-relation. *ApplicationRequirements* are further refined and detailed specified by functional system requirements and models.

The *Design Relation* between requirements outlined in the RE Artifact Model structure (Figure 12, chapter 4) is realized in the data model by the *IsJustifiedBy*-relation, the classification of *BusinessRequirements* into *BusinessGoals, QualityGoals* and *Features*, the classification of *ApplicationRequirements* into *UseCases*, *ArchitecturalConstrains, ModalConstraints* (state/mode requirements*), DTDConstraints* (data structure requirements) and the *Association*-relation to specify the constructed system models (right part of Figure C-2).

Analysis and Modeling by Functional System Views

The functional modeling concept of the RE Artifact Model analyzes and specifies requirements by basic system modeling views: *Process View, Structural View, Interaction View, Behavior View* and *Data View*. Figure C-3 illustrates this using the RM data model of REM.

The *Process View* hereby covers process and scenario modeling of business and system use processes. The focus is on analyzing and defining the main application tasks and its logic interaction with the overall system and the SuC. Together with the complementary modeling views, the main services or functions of the future system are derived and specified.

The Structural View is used to model the relevant actors, components and its interrelations. It models the relevant logic structure of the application domain such as the operational system environment or corresponding product-line components. The system boundaries will be derived and the required interfaces defined. Another objective of the Structural View is the refinement and hierarchical ordering of the required system services. (This is supported by the refinement tree of *UseCases* (*Super-* and *SubApplicationRequirements*) of the REM RM data model, see Figure C-2.)

The *Interaction View* models the communication between the system and its environment (actors, components) while performing the application processes and system services. This modeling technique is used as major tool to analyze and derive the system behavior, the necessary system data and specify the system interfaces. (See the *Process View* and *Process Analysis* above.)

The *Behavior View* models the required system behavior in terms of states and transitions. It is used to analyze modes of use and failure critical scenarios, and to derive a largely complete and save behavior specification of the required system.

The *Data View* specifies the overall data of the system. Major parts are the interface data and data structures definitions and the state defining variables and constraints of the system.

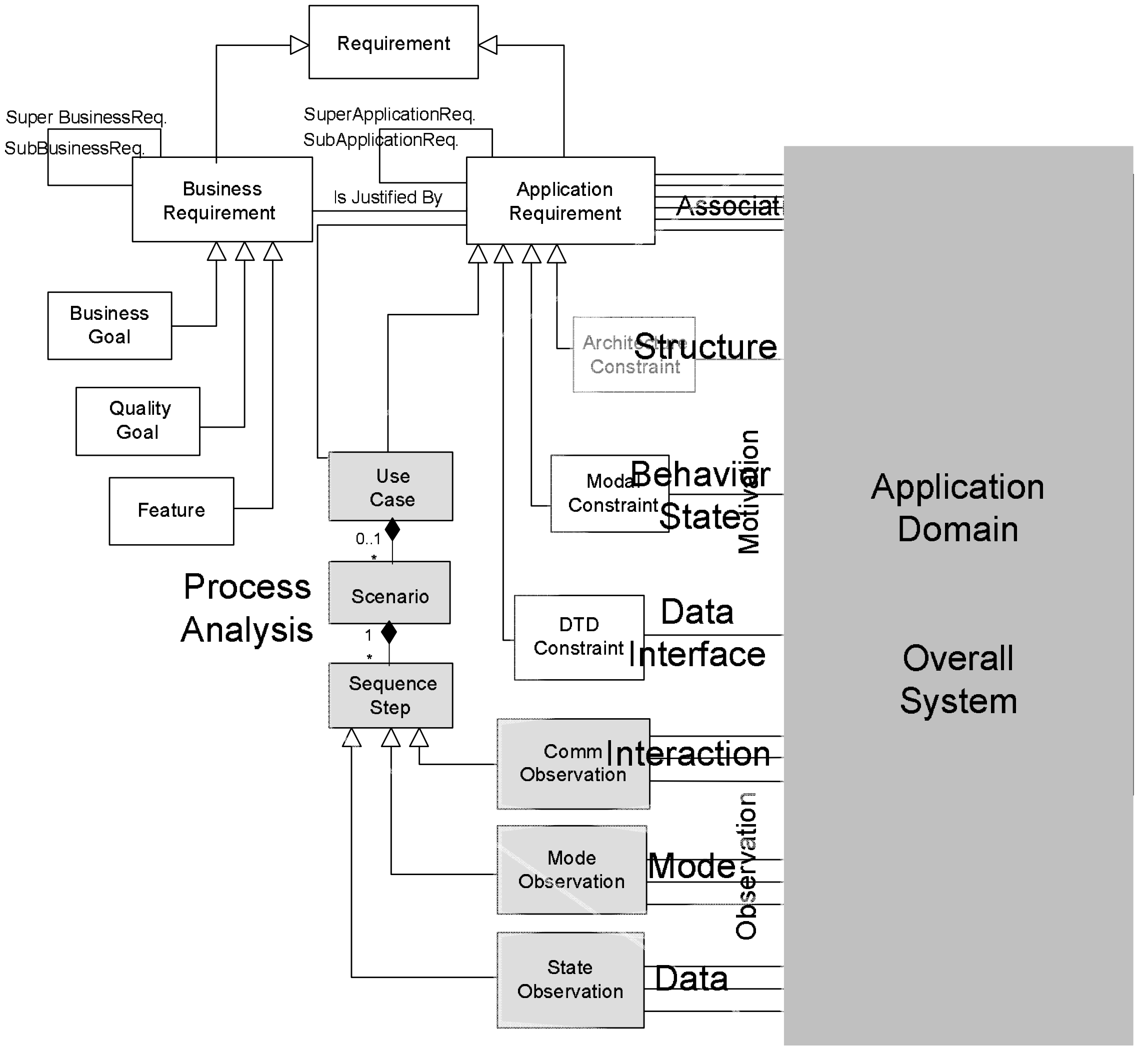


Figure C-3: Illustration of Process Analysis and System Views in REM

The RM data model in Figure C-2 and C-3 supports this *Process Analysis* steps by the specification concepts of *UseCases, Scenarios* and the systematic *Observation*-analysis of the single *SequenceSteps* within a *Scenario*. *CommObservations* analyze the scenario-related interaction between users, systems or components and support the corresponding construction and modeling (*Motivate*- and *Associate*-relation) of system components and communication *Interfaces* (see *Structural* and *Data View*). *ModeObservations* analyze the scenario-related use modes of the SuC and support the corresponding construction and modeling (*Motivate*- and *Associate*-relation) of system states and state conditions (see *Behavior* and *Data View*). *StateObservations* analyze the scenario-related states and interface data constraints of the involved system components and support the corresponding construction and modeling (*Motivate*- and *Associate*-relation) of state conditions and *Interfaces* data definitions (see *Behavior* and *Data View*).

Using the Construction Rules of the Underlying System Concept

By means of the projection of these view-based modeling of system requirements onto a common system concept a set of consistency constraints is defined. Figure C-4 shows the simplified data model of the system concept that underlies the specification tool AutoFocus [AutoFocus]. Figure C-5 shows the corresponding graphical description techniques of the tool: a system component tree (on the left), the (hierarchical) system structure modeling by *System Structure Diagrams (SSDs*), (hierarchical) state-transition-modeling by *State Transition Diagrams (STDs),* interaction modeling by *Extended Event Diagrams (EETs*) and the formal data type specification interface *(DTDs*) (on the right).

The system concept in Figure C-4 defines how the model elements of the different system views are related and how this consistency rules complement those view-based specifications. The rules can be systematically used to verify the consistency and completeness of the specified requirements and system models. If conflicting and missing requirements are discovered, they can be systematically treated and resolved by using the problem-oriented modeling rules of the Artifact Model and tracing them to the business goals and different stakeholder demands. This tracing enables negotiation and completion of requirements according to the business need of the actual product development.

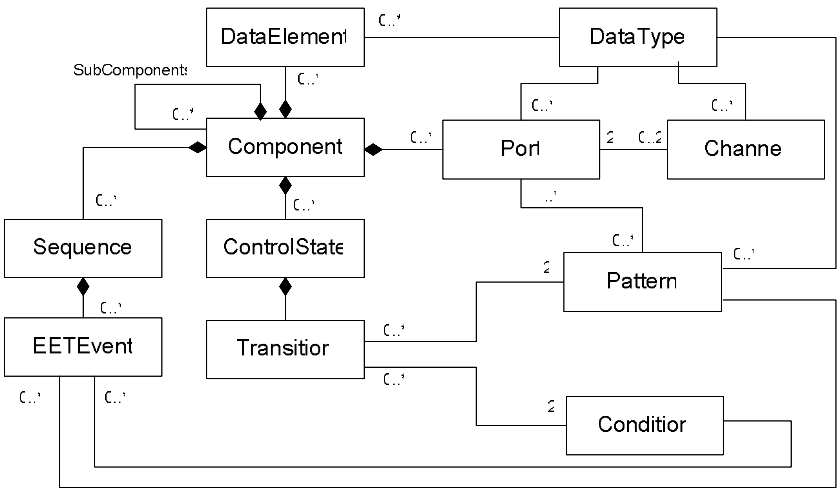


Figure C-4: System Concept data model of AutoFocus [AutoFocus]

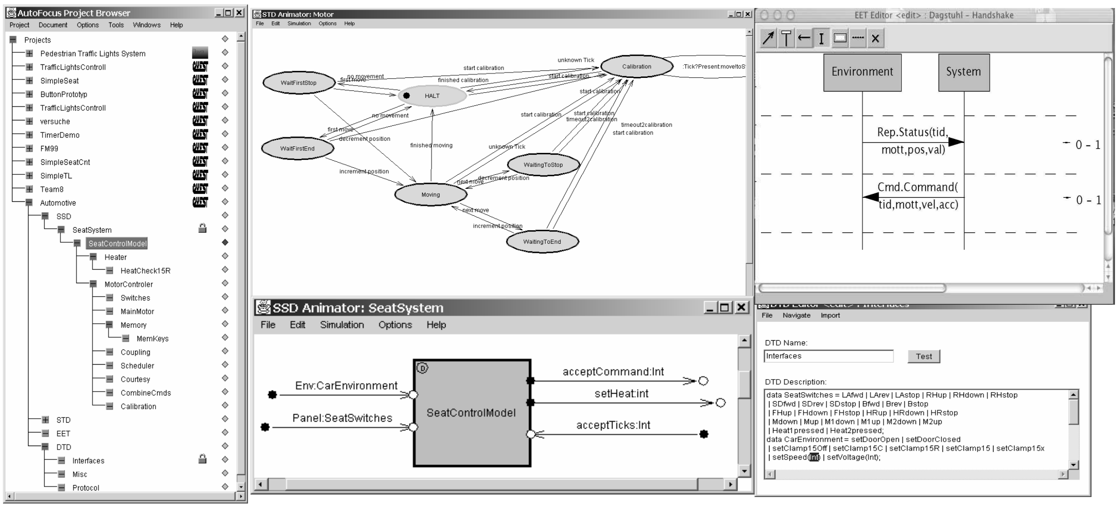


Figure C-5: Graphical system specification techniques of AutoFocus

Glossary, Terms and Definitions

The following list provides important definitions of terms used. It is essential that the reader of the document has skimmed through them to ensure a proper understanding of this document.

Agile process: Goal-oriented, flexibly adaptable development process based on core principles like customer-orientation, communication and simplicity – often based on prototyping and code-centric documentation.

Application: The use of a defined functionality or services of a system. Sometimes used as short-term “application system” for software systems that realize such a defined functionality or services.

Application system: A system that provides a set of general services for solving certain types of user problems or tasks.

Business rule: Business rules are operating principles about the product or system under consideration. They can lead to functional requirements that enforce them. [Wie99]

Constraint: Constraints are requirements that are precisely specified restrictions to the system design and development that narrow the potential solutions space. In opposite to general conditions, constraints must be concrete and measurably specified.

Context: The circumstances, situation, environment or domain in which a particular system exists or has to be considered.

Domain: An application area for a set of current or future application systems which share a set of common capabilities and data. [KC+90]

Feature: A prominent or distinctive user-/ customer-visible aspect, quality, or characteristic of a system [KC+90].

Functional system concept: A specification of the required behavior of a system in relation to its designed environment. It specifies the services that the system offers and their interaction behavior by the functional system views of modeling system use processes, interaction, (interface) data and states and transitions.

General condition: General conditions are requirements that state defaults to the development and design of a product/ system. The term ‘general’ refers to the underspecified character of those requirements. The term ‘general condition’ is often used synonymously to high-level non-functional requirements.

Meta-model:	A model that defines rules to build and verify certain models. Common description techniques to define and communicate meta-models are class diagrams, entity relationship diagrams or the OMG Standard Meta Object Facility (MOF) [MOF06].
Model:	An abstraction of complex facts, contexts or systems, that is build with the intention to capture/to understand them more easy or to make them workable by computers, often represented in terms of predefined structures.
Overall system:	Product or system in which the system under consideration has to be integrated.
Requirement:	(1) A condition or capability needed by a user to solve a problem or achieve an objective. (2) A condition or capability that must be met or possessed by a product or product component to satisfy a contract, standard, specification, or other formally imposed documents. (3) A documented representation of a condition or capability as in (1) or (2). [IEEE 610.12-1990]. Here it is, in particular, crucial, to distinguish between a stake holder request, which formulates a wish to be considered as a *potential* requirement, and a requirement, as a formally accepted and agreed obligation for the development.
High-level req.:	Requirements specifying the objective/goals (functional, quality) and conditions of a system development. They are inherently vague and underspecified and contain implications (requirements) on system functionality and quality.
Functional req.:	Requirements that specify the required behavior of a system/component in relation to its environment. They are usually specified by models of required system interaction.
Non-functional req.:	High-level requirements and further constraints/attribute on the functionality of the system under consideration.

Requirements-tracing:	Definition and tracing of relations/dependencies between requirements and to other artifacts in a development process (e. g. design or realization artifacts). Major objective is to confirm the realization of the requirements in the final product (forward tracing) and in return the compliance of the product with costumer and business needs (backward tracing). Thus, requirements-tracing supports verification and validation.
RE work product	A work result of RE tasks within a project to develop a product or product-line.
RE artifact	A work product that is described, reviewed, archived, and used as an input to a further tasks as part of the product development process, particularly the RE process.
Specification:	A specification is a precise description of a set of statements about the functionality and properties of a certain problem, product or system. In ideal case, it is a declarative description.
System specification:	A system specification is the description of a set of properties of a system; a system specification can be given in terms of models and in a formalized notion. The style of the specification and the choice of the model depend on the intention of its analyst/design engineers.
Stakeholder:	Person or group of persons that are affected by the development or the deployment of a system in any way. Stakeholders may include project members, suppliers, customers, end users, and others.
System:	A system is a delimited set of interrelated and interacting elements called components; it works as a coherent entity and shows a certain behavior. A system as a whole has system boundaries and a system environment. The environment is clearly separated of the system but consists of a family of systems again. If necessary, the components also could be seen as systems and further be analyzed; components form sub-systems.
User:	Either a person or an application (system) that operates and interacts with a system in order to perform a task [KC+90] using the system's functionality. Within this document the term user is used in the sense of human users only.
Validation	Validation checks and confirms that the product of a development project or intention will fulfill customer, user and business needs/requirements. In other words, validation ensures that "you built the right product."

Verification	Verification confirms that the result of a system development meets its specified requirements. Then it is called correct. In other words, verification ensures that "you built the product right."