Basics : the Requirements Engineering Process

Gregor v. Bochmann, University of Ottawa

Based on Powerpoint slides prepared by Gunter Mussbacher
with material from:
Requirements within the software development process

Figure 1.7 Step-wise quality assurance
What is the right system to build?

- How the customer explained it
- How the Project Leader understood it
- How the Analyst designed it
- How the Programmer wrote it
- How the Business Consultant described it
- How the project was documented
- What operations installed
- How the customer was billed
- How it was supported
- What the customer really needed
Table of Contents

- The Requirements Engineering Process
- Problem Domain and the System/Software-to-be
- Requirements Engineering: Main Activities

- The beginning is the most important part of the work.¹

¹ Plato, 4 B.C.
The Requirements Engineering Process
Figure 1-1  Relationship of several types of requirements information.
Notes on previous slide

- There needs to be an arrow from User requirements to System requirements. (The system has to be able to perform certain use cases. The same use cases must be supported by the software, therefore become Software requirements.)
- Business rules (including standards and regulations) are not only non-functional, they also include functional aspects (as shown by the arrows in the diagram).
RE process model

(suggested by Bray)

Again, this diagram shows

- RE activities (elicitation, analysis, specification, HMI design)
- subsequent design activity (internal design)
- RE documents (requirements, specification, HMI specification)

Important point:

Distinction between

- Problem domain (described by requ. doc.)
- System (to be built) (described by spec. doc.)

Note: one has to distinguish between current (problematic) version of the problem domain, and the projected future version which includes the system to be built.

Figure 2.1
Note: CRS – client requirements specification
Typical Layered Approach (V-shaped)

Source: Hull, Jackson, Dick: Requirements Engineering, 2004
• This looks like the waterfall process model, but this diagram describes a quite different situation.

• The layers correspond to step-wise refinement in terms of component decomposition.

• For instance, the transition from the first to the second layer is the typical RE process: one starts with the information from elicitation (shown in the first layer), that is, the problematic domain model, and one ends up with a proposal for a new system to be built (which is a component within the projected new domain model).

• **Important note:** The process of identification of the system to be built and defining its relationship with the new domain model (note that the environment of the system to be built may also be re-organized within the new domain model) is a kind of “design process” that requires creativity.

• The transitions to the lower layers in the diagram are similar processes (you may call them RE at a more detailed level or design processes)
Difference between RE and design??

• Much research towards automated SE
  • Compilers automatically generate machine code (correct in respect to program source code)
  • CASE tools automatically generate implementations of UML State Machine models (correct in respect to given model)
  • CASE tools automatically generate state machine models from a set of use case scenarios
    • E.g. PhD work of Dr. Somé
    • Tool for Live Sequence Charts by Dave - described in the book "Come, Let's Play: Scenario-Based Programming Using LSCs and the Play-Engine"
Harel’s “scenario-based programming” (1)

- Scenarios (use cases) are played into the tool, and may be played out for testing the recorded behavior model.

Fig. 2.4. Conventional system development
Main idea:

eliminate the design and implementation activities by providing efficient execution of behavior directly defined by the requirements.
Requirements and Modeling go together

• The systems engineering sandwich!

Comments on previous slide

• **Why combining RE with modeling?**
  • For analysis – models help to understand the problem domain
  • For documentation – models can be used for describing requirements (instead of solely using natural language)

• We will come back to various modeling approaches later in this course.
Back to the Sandwich – consider different levels of details

Requirements management

- Statement of need
- Stakeholder requirements
- System requirements
- Architectural design

Modeling and analysis

- e.g. Goal / Usage modeling
- e.g. Functional modeling
- e.g. Performance modeling

Source: Hull, Jackson, Dick: Requirements Engineering, 2004
**Benefits of Requirement Levels (Sandwich)**

**Principle of step-wise refinement:**
- Focus the attention on the big picture before addressing the details
- Reduce the number of changes by specifying at a lower level the requirements that will not affect the requirements at a higher level
- Promote the division of work

*This diagram [Lamsweerde] is another way to present this kind of (spiral) process*

*Figure 1.6  The requirements engineering process*
RE Process and Related Activities

<table>
<thead>
<tr>
<th>Why?</th>
<th>Identify Business Needs and Goals</th>
</tr>
</thead>
<tbody>
<tr>
<td>What?</td>
<td>Derive User &amp; Functional Requirements</td>
</tr>
<tr>
<td>How?</td>
<td>Design Solutions</td>
</tr>
<tr>
<td>Who?</td>
<td>Project Management Process</td>
</tr>
<tr>
<td>When?</td>
<td>Project Management Process</td>
</tr>
<tr>
<td>If-Then</td>
<td>Risk Management Process</td>
</tr>
<tr>
<td>Does It?</td>
<td>Quality Management Process</td>
</tr>
<tr>
<td>Where?</td>
<td>Component &amp; Configuration Management Process</td>
</tr>
</tbody>
</table>
Requirements engineering is a set of activities but not necessarily a separate phase.
The Problem Domain and the System/Software-to-be
Problem Domain

- The problem domain is the context for requirements
  - Part of the world within which the problem exists
  - Needs to be understood for effective requirements engineering

- Domain model
  - Set of properties assumed / believed to be true about the environment
  - Properties relevant to the problem
  - Problem domain requirements should hold in the proposed new version of the domain.

- Define the system requirements such that:
  - If the system that is built satisfies the system requirements and the environment satisfies the properties assumed for the environment, then the problem domain requirements will be satisfied.
  - In simple words: The system will behave as required if the assumptions hold.
Problem Domain and System-to-be

A domain model should be reusable
(Michael Jackson, 1995)

Diagram also showing activities [Bray]

Diagram showing existing and future situation [Lamsweerde]

Problem domain with system-to-be [Bray]
System interface and software interface

- System and software interface for a control system with embedded software:
  - Software interface: through input and output variables, for instance `measuredSpeed` (is read by program) and `doorState` (is set by program)
  - The system includes the software and I/O devices. Therefore the interface of the system with the environment are the monitored and controlled variables of the real world, for instance `trainSpeed` and `doorsClosed`.

Generic architecture of a control system including embedded software [Lamsweerde]
Software objects representing real objects

- The software (model) normally contains objects that represent objects in the system environment (e.g. the doorState variable represents the state of the doors in the train).
- Whether they represent the situation in the environment correctly, is another question (for the doorState variable, this may depend on the correct functioning of the door state sensing device).

**Better:** Problem domain requirements (if one considers the train to be the environment of the control system).

![Diagram](image)

**Figure 1.3** Phenomena and statements about the environment and the software-to-be [Lamsweerde]
Main Requirements Activities
Requirements Inception

• Start the process
  • Identification of business need
  • New market opportunity
  • Great idea

• Involves
  • Building a business case
  • Preliminary feasibility assessment
  • Preliminary definition of project scope

• Stakeholders
  • Business managers, marketing people, product managers...

• Examples of techniques
  • Brainstorming, Joint Application Development (JAD) meeting…
Requirements Elicitation (1)

• Gathering of information
  • About problem domain
  • About problems requiring a solution
  • About constraints related to the problem or solution

• Questions that need to be answered
  • What is the system?
  • What are the goals of the system?
  • How is the work done now?
  • What are the problems?
  • How will the system solve these problems?
  • How will the system be used on a day-to-day basis?
  • Will performance issues or other constraints affect the way the solution is approached?
Requirements Elicitation (2)

- Overview of different sources
  - Customers and other stakeholders
  - Existing systems
  - Documentation
  - Domain experts
  - More ...

- Overview of different techniques
  - Brainstorming
  - Interviews
  - Task observations
  - Use cases / scenarios
  - Prototyping
  - More ...
## Requirements Analysis

- The process of studying and analyzing the needs of stakeholders (e.g., customer, user) in view of coming up with a “solution”. Such a solution may involve:
  - A new organization of the workflow in the company.
  - A new system (system-to-be, also called solution domain) which will be used in the existing or modified workflow.
  - A new software to be developed which is to run within the existing computer system or involving modified and/or additional hardware.

- Objectives
  - Detect and resolve conflicts between requirements (e.g., through negotiation)
  - Discover the boundaries of the system / software and how it must interact with its environment
  - Elaborate system requirements to derive software requirements
Requirements Specification

• The invention and definition of the behavior of a new system (solution domain) such that it will produce the required effects in the problem domain

• Requirements Analysis has defined the problem domain and the required effects

• Specification Document
  • A document that clearly and precisely describes, each of the essential requirements (functions, performance, design constraints, and quality attributes) of the software and the external interfaces
  • Each requirement being defined in such a way that its achievement is capable of being objectively verified by a prescribed method (e.g., inspection, demonstration, analysis, or test)
  • Different guidelines and templates exist for requirements specification
Validation and verification

- Both help ensure delivery of what the client wants
- Need to be performed at every stage during the process

Validation: checks that the **right product is being built** (refers back to stakeholders – main concern during RE)

Verification: checks that the **product is being built right**

- During design phase: refers back to the specification of the system or software requirements
- During RE: mainly checking consistency between different requirements, detecting conflicts

Techniques used during RE

- Simple checks
- Formal Review
- Logical analysis
- Prototypes and enactments
- Design of functional tests
- Development of user manual
Requirements Management

- Necessary to cope with changes to requirements
- Requirements change is caused by:
  - Business process changes
  - Technology changes
  - Better understanding of the problem

- Traceability is very important for effective requirements management

Elicitation notes

Goals rationale

Requirements document
1.1 XXXX 
.... because 
1.2 YYYY

Design document
....due to 
requirement 1.2
Requirements Documents

- Vision and Scope Document
- Elicitation notes: (Raw) requirements obtained through elicitation; often unstructured, incomplete, and inconsistent
- (Problem domain) Requirements document
- System requirements document
- Software requirements document
  - The software is normally part of a system that includes hardware and software. Therefore, the software requirements are normally part of the system requirements.
- Note: System and Software requirements may exist in several versions with different levels of details, such as
  - User (customer) requirements: Statements in natural language plus diagrams of the services the system provides and its operational constraints; written for customers
  - Detailed requirements: A structured document setting out detailed descriptions of the system services; often used as a contract between client and contractor. This description can serve as a basis for a design or implementation; used by developers.
Types of Requirements Documents

Two extremes:

• An informal outline of the requirements using a few paragraphs or simple diagrams
  • This is called the requirements definition
• A long list of specifications that contain thousands of pages of intricate requirements describing the system in detail
  • This is called the requirements specification

• Requirements documents for large systems are normally arranged in a hierarchy
The Requirements Analyst\(^1\)

- Plays an essential communication role
  - Talks to users: application domain
  - Talks to developers: technical domain
  - Translates user requirements into functional requirements and quality goals

- Needs many capabilities
  - Interviewing and listening skills
  - Facilitation and interpersonal skills
  - Writing and modeling skills
  - Organizational ability

- RE is more than just modeling…
  This is a social activity!

\(^1\) Karl Wiegers, In Search of Excellent Requirements
For More Information

- Simson Garfinkel, History's Worst Software Bugs, Wired News, 2005
  [http://www.wired.com/news/technology/bugs/0,2924,69355,00.html]
- INCOSE Requirements Working Group
  [http://www.incose.org/practice/techactivities/wg/rqmts/]
- Tools Survey: Requirements Management (RM) Tools
  [http://www.incose.org/productspubs/products/rmsurvey.aspx]
  [http://www.volere.co.uk/tools.htm]
- Requirements Engineering Conference
  [http://www.requirements-engineering.org/]

SEG3101 (Fall 2010)  Basics – the RE process.
Main References

Jeremy Dick, Elizabeth Hull, Ken Jackson: Requirements Engineering, Springer-Verlag, 2004
Soren Lauesen: Software Requirements - Styles and Techniques, Addison Wesley, 2002
Ian K. Bray: An Introduction to Requirements Engineering, Addison Wesley, 2002
Karl E. Wiegers: Software Requirements, Microsoft Press, 2003
CHAOS Report, Standish Group