Chapter 5
Requirements Elicitation I

Organizational Requirements Engineering

Prof. Dr. Armin B. Cremers
Sascha Alda
Overview

- General view on requirements elicitation
- Processes of requirements elicitation (and analysis)
- Elicitation Techniques
  - Scenarios
  - Interviews
  - Observation
  - Prototyping
- From scenarios to use cases
  - Identifying actors
  - Best practice for modeling use cases
  - Refinement of use cases
- Conclusions
Software Development Process: Classify Requirements Elicitation

Use Case Model

Requirements Elicitation

Analysis
System Design
Object Design
Implementation
Testing

Expressed in Terms of
Structured by
Realized by
Implemented by
Verified by

Application Domain Objects
Subsystems
Solution Domain Objects
Source Code
Test Cases

Armin B. Cremers, Sascha Alda
Organizational Requirements Engineering
Requirements Elicitation
First view

- Encompass all activities involved in discovering the requirements of a system
- System developers and engineers work in close relationship with customer and end-users to
  - Find out more about the problem to be solved
  - To describe the functionality of the system
  - Performance of the system
  - Hardware constraints … and so forth
- Not just a simple process about fishing for requirements, but a highly complex process:
  - Customer rarely have a clear picture of their requirements
  - Different people have conflicting requirements
Requirements Elicitation
Further aspects

- Business environment in which the elicitation process takes place is constantly changing
  - Importance and relevance some requirements may change
  - New requirements may result from new stakeholders
  - End-user may change their jobs

- Many different elicitation processes can be found
- Here: concentration on general-purpose process models
Changing Requirements and Costs

- **Traditional view**
  - Costs of changes develop exponentially due to time
  - Complexity of software increases over project time (and phases)

- **Alternative view in Agile Development (e.g. XP)**
  - Goal: Linear development of costs
  - Changes can be done everytime during the project → higher flexibility
Components of requirements elicitation: Basic Elicitation activities

- Application domain understanding
  - knowledge of the general area where the system is applied
- Problem understanding
  - details of the specific customer problem where the system will be applied
- Business understanding
  - understand how systems interact and contribute to overall business goals
- Understanding the needs and constraints of system stakeholders
  - understand, in detail, the specific needs of people who require system support in their work
Elicitation, analysis and negotiation: Yet another view on RE
The requirements elicitation process: Elicitation stages

- **Objective setting**: establish the overall organizational objectives: why is the system necessary?
- **Background of knowledge acquisition**: gather and understand more background information about the system
The requirements elicitation process: Elicitation stages

<table>
<thead>
<tr>
<th>Establish Objects</th>
<th>Understand Background</th>
<th>Organize Knowledge</th>
<th>Collect Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Business Goals</td>
<td>Organizational structure</td>
<td>Stakeholder Identification</td>
<td>Stakeholder requirements</td>
</tr>
<tr>
<td>Problem to be solved</td>
<td>Application Domain</td>
<td>Goal Priorization</td>
<td>Domain requirements</td>
</tr>
<tr>
<td>System Constraints</td>
<td>Existing Systems</td>
<td>Domain knowledge filtering</td>
<td>Organizational Requirements</td>
</tr>
</tbody>
</table>

- **Knowledge organization**: organize, prioritize and collate the large amount of data collected in the previous phases
- **Stakeholder requirements collection**: involve system stakeholders to discover their requirements.
Necessity checking
- Sometimes requirements don’t contribute to the business goals of the organization or to the specific problem to be addressed by the system

Consistency and completeness checking
- Cross-checking for consistency and completeness (no contradictions, no services or constraints are missed out)

Feasibility checking
- The context of the budget and schedule
Requirements Analysis and Negotiation: Negotiation

- Requirements discussion
  - Requirements highlighted as problematical are discussed
  - the stakeholders involved present their views about the requirements

- Requirements prioritization
  - Identification of critical requirements

- Requirements agreement
  - A compromised set of requirements are agreed
  - changes to some of the requirements
Process of Requirements Elicitation: Products of Requirements Process

- Problem Statement
- Requirements specification: functional and non-functional requirements
- Analysis Model: dynamic model, object model
- Requirements Elicitation
- Requirements Analysis
The Goal: Analysis Model (vs. Requirements Specification)

- Both models focus on the requirements from the user’s view of the system.
- Requirements specification uses natural language (derived from the problem statement)
- The analysis model uses formal (Z, pi-calculus) or semi-formal notation (for example, a graphical language like UML)
  - Formal notations encompass an exact mathematical syntax and semantic
- The starting point is the problem statement
Starting with the Problem Statement

- The problem statement is developed by the client as a condensed description of the requirements that should be addressed by the system
- Describes the problem that should be solved
- It describes “what” is needed, not “how” it should be reached
Starting with the Problem Statement: Ingredients

- Current situation: The Problem to be solved
  - A few pages
- Description of one or more scenarios
- Some initial requirements
  - Functional and Non-functional requirements
  - No complete description
- Project Schedule
  - Major milestones that involve interaction with the client including deadline for delivery of the system
- Target environment
  - The environment in which the delivered system has to perform a specified set of system tests
- Client Acceptance Criteria
  - Criteria for the system tests
Starting with the Problem Statement: Current Situation - Problem To Be Solved

- There is a problem in the current situation
  - Examples:
    - The response time in a travel booking system is far too slow
    - There have been illegal attacks to the system
- A change either in the application domain or in the solution domain has appeared
  - Change in the application domain
    - A new function (business process) is introduced into the business
    - Example: A function is provided for credit payment with fingerprint as authorization
  - Change in the solution domain
    - A new solution (technology enabler) has appeared
    - Example: New standards (implementation) for secure network communication
Example: Library System

- Idea: A Library Management System should be designed. Information on books, CDs, DVDs, Journals, etc. can be stored and retrieved.

- Possible Requirements:
  - Searching by Title, Author, and/or ISDN should be possible
  - User Interface should be web-based (accessible via WWW Browser)
  - At least 20 transactions per seconds should be possible
  - All services should be available within 10 minutes
  - Users have no access to personal data of other users
Process of Requirements Elicitation:
The Requirements Elicitation Cycle (Brügge)

- Observing users
- Interviewing users and clients
- As-Is Scenarios
- Visionary Scenarios
- Use Cases + Refinements
- Prototypes

Validation

Tests

Stable Requirements Specification
(System Specification)
- Functional Requirements
- Non-Functional Requirements
- Use Cases
- Scenarios
**Process of Requirements Elicitation:**
Activities during Requirements Elicitation

- Identifying Actors
  - Types of users, roles, external systems

- Identifying Scenarios
  - Interactions between users and the systems (one possible case)
  - → *Later on in this lesson*

- Identifying Use Cases
  - Abstractions of Scenarios
    - (Many possible cases)
  - → *Next Lesson*

- Refining Use Cases
  - Refinements, adding exceptions, etc.

- Identifying Relationships among Use Cases

- Identifying Non-Functional Requirements
  - Security issues, Performance, etc.
Process of Requirements Elicitation: How to elicit Requirements?

- Sources of information
  - Documents about the application domain
  - Manual and technical documents of legacy systems

- User Participation
  - Interviews
    - Closed Interviews: User answer a predefined set of questions
    - Open Interviews: No predefined agenda
  - Work Practice
  - User Observation (next lesson)

→ Describing Scenarios
Process of Requirements Elicitation: The Requirements Elicitation Cycle

- Observing users
- Interviewing users and clients
- As-Is Scenarios
- Visionary Scenarios
- Use Cases + Refinements
- Prototypes
- Tests
- Validation

Stable Requirements Specification (System Specification):
- Functional Requirements
- Non-Functional Requirements
- Use Cases
- Scenarios
Elicitation techniques - Idea

- Specific techniques which may be used to collect knowledge about system requirements
- This knowledge must be structured
  - Partitioning - aggregating related knowledge
  - Abstraction - recognizing generalities
  - Projection - organizing knowledge from several perspectives
- Requirements elicitation is a cooperative process involving requirements engineers and system stakeholders. Problems:
  - Not enough time for elicitation
  - Inadequate preparation by engineers
  - Stakeholders are unconvinced of the need for a new system
Selection Criteria

- System to be created (I)
  - Greenfield Engineering
  - Reengineering
  - Interface Engineering

- System to be created (II)
  - Highly interactive (Cooperation Support System)
  - Specific applications like Games

- Budget/Time

- Degree of User Participation
  - Time
  - Experience of users

- .... (many more)
Elicitation techniques - Overview

This lesson:
- Scenarios
- Interviews

Next lesson:
- Soft systems methods
- Observations and social analysis
- Prototyping
Interviews
Essentials

- Probably the most common technique of requirements elicitation.
- Interviewers must be open-minded and should not approach the interview with pre-conceived notions about what is required.
- Stakeholders must be given a starting point for discussion:
  - a question
  - a requirements proposal
  - an existing system
- Interviewers must be aware of organizational politics:
  - Some requirements may not be discussed because of their political implications.
- Interviews with different stakeholders:
  - Different perspectives
  - global understanding of their requirements
- Interviews no good way for understanding concepts of application domain.
Interviews
Different Techniques

● Structured (closed) interviews
  ◆ Stakeholders answer a predefined set of questions
  ◆ Easy to analyze (+)
  ◆ Well-formed questions generate well-formed answers (you have to know your goals) (+)
  ◆ Knowledge about what and how to ask (-)

● Non-structured (open) interviews
  ◆ No predefined agenda
  ◆ Generating new ideas (experimental, brainstorming) (+)
  ◆ Sometimes hard to handle (dynamics of discussion) (-)

● In practice: mixed interview types are normal
Interviews
Written vs. oral interviews, group vs. person

- Oral interviews:
  - possibility to discussion (+)
  - interviewer may influence interviewee (-)

- Written interviews
  - problems in understanding (-)
  - already transcribed, thus easy to analyze (+)

- Interviewing a single person:
  - individual opinions (+)

- Interviewing a group of people:
  - Involvement of many perspectives
  - Developer need experience to moderate (-)
  - Sometimes single or silent opinions are not noticed (-)
  - Independent Moderator can mediate between group
Interviews
Good practices (selection)

- Prepare some initial questions → good entry point
- Do not press the interviewee through the questionnaire
- Restrict the time frame for the questionnaire (approx. 1 hour)
  - Announce the estimated time for the interview
- Short introduction for the purpose of the interview
- Ensure anonymity (if necessary)
- Make notes to the answers
  - Explain the purpose of the records (reduces potential fear!)
- Do not interrupt the interviewee’s flow of words
- Allow people to refuse a question (don’t insist on answers!)
- Announce feedback (evaluation) at the end
Interviews

Transcription

- Transcription is the compilation of human communication into scripture, mostly based on voice or written recording.
- Transcription systems are rule types that exactly determines how spoken language is to be compiled. Incorporates different attributes:
  - Loudness of speech
  - Speech intermission
  - Gesture, facial expression
  - Emphasis of special words or phrases
  - Repeat of phrases
  - External factors
- Software support can be utilized (syncWRITER)
- Interview should be transcribed by second person
Interviews IV: Different Goals

- During elicitation (early)
  - Understanding role of interviewee within organization
  - Understanding the work context
  - Getting requirements on new system
  - Reviewing first pass of scenarios (exercise)

  **Goal: Description of complete scenarios (next...)**

- During analysis
  - Discussing use cases with client and users
  - Correction and refinement (requirements and functionality)

  **Goal: Getting complete use cases (next lesson)**
Scenarios - Overview 1

Motivation (Observation):

- System stakeholders find it more intuitive to reason about concrete examples rather than abstract descriptions of the functions provided by a system (use cases).

Solution: Scenario

- “A narrative description of what people do and experience as they try to make use of computer systems and applications” [M. Carrol, Scenario-based Design, Wiley, 1995]
- A concrete, focused, informal description of a single feature of the system used by a single actor.
- Discovering scenarios exposes possible system interactions and reveals system facilities which may be required.
Scenarios - Overview 2

- Scenarios are stories which explain how a system might be used. They should include:
  - a description of the system state before entering the scenario
  - the normal flow of events in the scenario
  - exceptions to the normal flow of events
  - information about concurrent activities
  - a description of the system state at the end of the scenario

- Scenarios can have many different uses during the software lifecycle:
  - **Requirements Elicitation**: As-is scenario, visionary scenario
  - **Client Acceptance Test**: Evaluation scenario
  - **System Deployment**: Training scenario.
Scenarios: Different Types

- **As-is scenario**
  - Used in describing a current situation
  - Usually used in re-engineering projects
  - The user describes the system

- **Visionary scenario**
  - Used to describe a future system
  - Usually used in Greenfield engineering and reengineering projects
  - Can often not be done by the user or developer alone
    - brainstorming sessions, future workshop

- **Evaluation scenario**
  - User tasks against which the system is to be evaluated

- **Training scenario**
  - Step by step instructions that guide a novice user through a system
Scenarios: How do we find scenarios?

- Interviews with stakeholder
- Don’t expect the client to be verbal, if the system does not exist (Greenfield engineering)
- Don’t wait for information even if the system exists
- Developer and user profit from creating scenario both-way:
  - You help the client to formulate the requirements
  - The client helps you to understand the requirements
  - The requirements evolve and become more obvious while the scenarios are being developed
Scenarios:
Possible questions in an interview (visionary)

- What are the primary tasks that the system needs to perform?
- How do you currently perform your primary task?
- Do you know about any kind of system or service that already fulfills some task?
- What data will the (main) actor create, store, change, remove or add in the system?
- Are there other actors in the system (explain the term actor!)
- Do the actors need assistance during carrying out their tasks?
- What external changes does the system need to know about?
- What changes or events will the actor of the system need to be informed about?
- What kind of exceptions can you suggest?
- Can actors interrupt a sequence of interaction? What happens, if so?
- What about extra-ordinary events and tasks?
Scenarios:
Heuristics for finding Scenarios

- However, don’t rely on questionnaires alone.
- Insist on task observation if the system already exists (interface engineering or reengineering)
  - Ask to speak to the end user, not just to the software contractor
  - Expect resistance and try to overcome it
Scenarios:
Example - Accident Management System

Your Task (Problem Statement):

- Build a requirements model for a system that allows to report fire incidents. It should be able to report incidents for many types of buildings and things.

The approach: Start with single Scenario, e.g. “Warehouse in fire”. Interview Guideline:

- What do you need to do if a person reports “Warehouse on Fire?”
- Who is involved in reporting an incident?
- What does the system do, if no police cars are available? If the police car has an accident on the way to the “cat in a tree” incident?
- Can the system cope with a simultaneous incident report “Warehouse on Fire?”
- What do you need to do if the “Warehouse on Fire” turns into a “Cat in the Tree”?
Scenario:
Example - Warehouse on Fire

- **Bob**, driving down main street in his patrol car notices smoke coming out of a warehouse. His partner, **Alice**, reports the emergency from her car by using the **SYSTEM**.

- **Alice** enters the address of the building, a brief description of its location (i.e., north west corner), and an emergency level. In addition to a fire unit, she requests several paramedic units on the scene given that area appear to be relatively busy. She confirms her input and waits for an acknowledgment.

- **John**, the Dispatcher, is alerted to the emergency by a beep of his workstation. He reviews the information submitted by Alice and acknowledges the report. He allocates a fire unit and two paramedic units to the Incident site and sends their estimated arrival time (ETA) to Alice.

- **Alice** received the acknowledgment and the ETA.
Scenarios:
Observations about “Warehouse on Fire”

- Concrete scenario
  - Describes a single instance of reporting a fire incident.
  - Does not describe all possible situations in which a fire can be reported.

- Participating actors
  - Bob, Alice and John
From Scenarios to use cases
First pass

- Use case: an abstraction of possible coherent scenarios
- Scenario: a single example of a scenario
  → instance of a use case!

Example:

Use Case
“ReportFireIncident”

Scenario
“Warehouse on Fire”

Scenario
“Flat on Fire”

Scenario
“Car on Fire”

= “ReportFireIncident”
Summary
(Requirements Elicitation Overview)

● The goal is a sound model representing the requirements of the system seen from the user’s perspective

● First steps are:
  ♦ Write the Problem Statement
  ♦ Elicit Requirements

● First step of elicitation is understanding scenarios

● Requirements elicitation is a cyclic process