Chapter 6

Requirements Engineering Processes
Processes used to discover, analyze, and validate system requirements

Requirements engineering processes
The processes used in requirement engineering vary widely depending on the application domain, the people involved and the organization developing the requirements. Nevertheless, there are four activities common to all:

• Requirements elicitation
• Requirements analysis
• Requirements validation
• Requirements management

The requirements engineering process

Feasibility studies
Designed to determine whether or not the proposed system
• will contribute to organizational objectives,
• can be built with current technology within budget, and
• can be integrated with other systems in use.

Feasibility study (continued)

- It involves information assessment, information collection and report writing.
- Questions for people in the organization
  - What if the system wasn’t implemented?
  - What are current process problems?
  - How will the proposed system help?
  - What will be the integration problems?
  - Is new technology needed? What skills?
  - What must be supported by the proposed system?

Elicitation and analysis

- It involves the technical staff working with the customers to understand the application domain, and to determine the services that the system should provide as well as the system's operational constraints.
- May involve end-users, managers, engineers involved in maintenance, domain experts, trade unions, etc. They are called stakeholders.
Problems of requirements analysis

- Stakeholders don’t know what they really want.
- Stakeholders express requirements in their own terms.
- Stakeholders may have conflicting requirements.
- Organizational and political factors may influence the system requirements.
- The requirements may change during the analysis process due to emergence of new stakeholders or change in business environment.

Elicitation and analysis activities

- Domain understanding
- Requirements collection
- Classification
- Conflict resolution
- Prioritization
- Requirements checking

Elicitation and analysis process

Viewpoint-oriented elicitation

- Stakeholders represent different ways of looking at a problem
- This multi-perspective analysis is important as there is no single correct way to analyze system requirements

Banking ATM system

- The example used here is an automated-teller-machine system which provides some automated banking services.
- It is a simple system that offers some services to customers of the bank who own the system and a narrower range of services to other customers.
- Services include cash withdrawal, message passing (send a message to request a service), ordering a statement and transferring funds.

Different viewpoints toward an ATM

- Bank customers
- Representatives of other banks
- Hardware and software maintenance engineers
- Marketing department
- Bank managers and counter staff
- Database administrators and security staff
- Communications engineers
- Personnel department
Types of viewpoint

- **Data sources or sinks**
  Viewpoints are responsible for producing or consuming data. Analysis involves checking that data is produced and consumed and that assumptions about the source and sink of data are valid.

- **Representation frameworks**
  Viewpoints represent particular types of system model. These may be compared to discover requirements that would be missed using a single representation. Particularly suitable for real-time systems.

- **Receivers of services**
  Viewpoints are external to the system and receive services from it. Most suited to interactive systems.

External viewpoints

- **Viewpoints are a natural way to structure requirements elicitation process**
- **It is relatively easy to decide if a viewpoint is valid**
- **Viewpoints and services may be used to structure non-functional requirements**

Method-based analysis

- **A widely used approach, it depends on the application of a structured method to understand the system.**
- **Methods have different emphases. Some are designed for requirements elicitation, others are close to design methods**
- **A viewpoint-oriented method (VORD) is used as an example here. It also illustrates the use of viewpoints.**

The VORD method

- **Viewpoint identification**
  Discover viewpoints which receive system services and identify the services provided to each viewpoint
- **Viewpoint structuring**
  Group related viewpoints into a hierarchy. Common services are provided at higher-levels in the hierarchy
- **Viewpoint documentation**
  Refine the description of the identified viewpoints and services
- **Viewpoint-system mapping**
  Transform the analysis to an object-oriented design

VORD process model

<table>
<thead>
<tr>
<th>Viewpoint template</th>
<th>Service template</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Reference:</strong> The viewpoint name.</td>
<td><strong>Reference:</strong> The service name.</td>
</tr>
<tr>
<td><strong>Attributes:</strong> Attributes providing viewpoint information.</td>
<td><strong>Rationale:</strong> Reason why the service is provided.</td>
</tr>
<tr>
<td><strong>Events:</strong> A reference to a set of scenarios describing how the system reacts to viewpoint events.</td>
<td><strong>Specification:</strong> Reference to a set of service specifications. These may be expressed in different notations.</td>
</tr>
<tr>
<td><strong>Services:</strong> A reference to a set of service descriptions.</td>
<td><strong>Viewpoints:</strong> List of viewpoint names receiving the service.</td>
</tr>
<tr>
<td><strong>Sub-VPs:</strong> The names of sub viewpoints.</td>
<td><strong>Non-functional requirements:</strong> Reference to a set of non-functional requirements which constrain the service.</td>
</tr>
<tr>
<td><strong>Provider:</strong> Reference to a set of system objects which provide the service.</td>
<td></td>
</tr>
</tbody>
</table>
Viewpoint identification

Query balance
Get transactions
Cash withdrawal
Transaction log
Remote software upgrade
Order cheques
Invasion
Print
Software site
Bank teller
Foreign customer
Hardware maintenance
Message passing
Funds transfer
Card validation
Reliability
Update account
Account holder
User interface
Machine supplies
System costs
Message log
Transaction log
Manager
Customer database
Get transactions
Customer

Viewpoint service information

ACCOUNT HOLDER
Service list
Withdraw cash
Query balance
Order cheques
Send message
Transaction list
Order statement
Transfer funds

FOREIGN CUSTOMER
Service list
Withdraw cash
Query balance

BANK TELLER
Service list
Run diagnostics
Add cash
Add paper
Send message

Viewpoint data/control

ACCOUNT HOLDER
Control input
Start transaction
Cancel transaction
End transaction
Select service

Data input
Card details
PIN
Amount required
Message

Viewpoint hierarchy

Services:
Query balance
Withdraw cash

Services:
Order cheques
Send message
Transaction list
Order statement
Transfer funds

Add VP
Customer
Bank staff
Account holder
Foreign customer
Teller
Manager
Engineer
Manager
Teller

Customer/cash withdrawal templates

Reference:
Customer
description:

Attributes:
Account number
PIN

Events:
Start transaction
Cancel transaction
End transaction
Select service

Services:
Cash withdrawal
Balance inquiry

Sub-VPs:
Account holder
Foreign customer

Reference:
Cash withdrawal
description:

Rationale:
To improve customer service and reduce paperwork

Specification:
Users choose this service by pressing the cash withdrawal button. They then enter the amount required. This is confirmed and, if funds allow, the balance is delivered.

VPs:
Customer

Non-funct. requirements:
Deliver cash within 1 minute

Provider:
Filled in later

Scenarios

- Scenarios are descriptions of how a system is used in practice.
- They are helpful in requirements elicitation as people can relate to these more readily than abstract statement of what they require from a system.
- Scenarios are particularly useful for adding detail to an outline requirements description.
Scenario descriptions

- System state at the beginning of the scenario
- Normal flow of events in the scenario
- What can go wrong and how this is handled
- Other concurrent activities
- System state on completion of the scenario

Event scenarios

- Event scenarios may be used to describe how a system responds to the occurrence of some particular event such as ‘start transaction’
- VORD includes a diagrammatic convention for event scenarios.
  - Data provided and delivered
  - Control information
  - Exception processing
  - The next expected event

Event scenario - start transaction

Notation for data and control analysis

- Ellipses, data provided from or delivered to a viewpoint
- Control information enters and leaves at the top of each box
- Data leaves from the right of each box
- Exceptions are shown at the bottom of each box
- Name of next event is in box with thick edges

Exception description

- Most methods do not include facilities for describing exceptions
- In this example, exceptions are
  - Timeout. Customer fails to enter a PIN within the allowed time limit
  - Invalid card. The card is not recognized and is returned
  - Stolen card. The card has been registered as stolen and is retained by the machine

Use cases

- Use-cases are a scenario based technique in the UML which identify the actors in an interaction and which describe the interaction itself
- A set of use cases should describe all possible interactions with the system
- Sequence diagrams may be used to add detail to use-cases by showing the sequence of event processing in the system
Social and organizational factors

- Software systems are used in a social and organizational context. This can influence or even dominate the system requirements
- Social and organizational factors are not a single viewpoint but are influences on all viewpoints
- Good analysts must be sensitive to these factors but currently no systematic way to tackle their analysis

Example

- Consider a system which allows senior management to access information without going through middle managers
  - Managerial status. Senior managers may feel that they are too important to use a keyboard. This may limit the type of system interface used
  - Managerial responsibilities. Managers may have no uninterrupted time where they can learn to use the system
  - Organizational resistance. Middle managers who will be made redundant may deliberately provide misleading or incomplete information so that the system will fail

Ethnography

- It is an observational technique that can be used to understand social and organizational requirements
- An analyst immerses himself in the working environment where the system will be used
- The day-to-day work is observed and notes made of the actual tasks in which participants are involved
- The value of ethnography is that it helps discover implicit system requirements

Focused ethnography

- Developed in a project studying the air traffic control process
- Combines ethnography with prototyping
- Prototype development results in unanswered questions which focus the ethnographic analysis
- Problem with ethnography is that it studies existing practices which may have some historical basis which is no longer relevant
Ethnography and prototyping

Scope of ethnography
- Requirements that are derived from the way that people actually work rather than the way in which process definitions suggest that they ought to work
- Requirements that are derived from cooperation and awareness of other people’s activities

Requirements validation
- Concerned with demonstrating that the requirements define the system that the customer really wants
- Requirements error costs are high so validation is very important
  - Fixing a requirements error after delivery may cost up to 100 times the cost of fixing an implementation error

Requirements checking
- Validity. Does the system provide the functions which best support the customer’s needs?
- Consistency. Are there any requirements conflicts?
- Completeness. Are all functions required by the customer included?
- Realism. Can the requirements be implemented given available budget and technology?
- Verifiability. Can the requirements be checked?

Requirements validation techniques
- Requirements reviews
  - Systematic manual analysis of the requirements
- Prototyping
  - Using an executable model of the system to check requirements. Covered in Chapter 8
- Test-case generation
  - Developing tests for requirements to check testability
- Consistency analysis
  - Checking the consistency of a structured requirements description

Requirements reviews
- Regular reviews should be held while the requirements definition is being formulated
- Both client and contractor staff should be involved in reviews
- Reviews may be formal (with completed documents) or informal.
Review checks

- **Verifiability.** Is the requirement realistically testable?
- **Comprehensibility.** Is the requirement properly understood?
- **Traceability.** Is the origin of the requirement clearly stated?
- **Adaptability.** Can the requirement be changed without a large impact on other requirements?

Automated consistency checking

Requirements management

- It is the process of managing changing requirements during the requirements engineering process and system development
- Requirements are inevitably incomplete and inconsistent
  - New requirements emerge during the process as business needs change and a better understanding of the system is developed
  - Different viewpoints have different requirements and these are often contradictory

Requirements change

- The priority of requirements from different viewpoints changes during the development process.
- System customers may specify requirements from a business perspective that conflict with end-user requirements.
- The business and technical environment of the system changes during its development

Enduring and volatile requirements

- Enduring requirements. Stable requirements derived from the core activity of the customer organization. E.g. a hospital will always have doctors, nurses, etc. May be derived from domain models
- Volatile requirements. Requirements which change during development or when the system is in use. In a hospital, requirements derived from health-care policy

Classification of requirements

- **Mutable requirements**
  - Requirements that change due to the system’s environment
- **Emergent requirements**
  - Requirements that emerge as understanding of the system develops
- **Consequential requirements**
  - Requirements that result from the introduction of the computer system
- **Compatibility requirements**
  - Requirements that depend on other systems or organizational processes
Requirements management planning

Plan for:
- Requirements identification: How requirements are individually identified?
- A change management process: How to assess the impact and cost of changes?
- Traceability policies: What relationships among requirements and system design need to be maintained?
- CASE tool support

Traceability

Three types of traceability information may be maintained:
- Source traceability
  - Links from requirements to stakeholders who proposed these requirements
- Requirements traceability
  - Links between dependent requirements
- Design traceability
  - Links from the requirements to the design

A traceability matrix

<table>
<thead>
<tr>
<th>Req</th>
<th>1.1</th>
<th>1.2</th>
<th>1.3</th>
<th>2.1</th>
<th>2.2</th>
<th>3.1</th>
<th>3.2</th>
<th>3.3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>U</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.2</td>
<td></td>
<td>U</td>
<td>R</td>
<td>U</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.3</td>
<td></td>
<td>R</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.1</td>
<td></td>
<td>R</td>
<td></td>
<td>U</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.2</td>
<td></td>
<td></td>
<td></td>
<td>U</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.1</td>
<td>R</td>
<td></td>
<td>U</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.2</td>
<td></td>
<td></td>
<td></td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

CASE tool support

- Requirements storage
  - Requirements should be managed in a secure, managed data store
- Change management
  - The process of change management is a workflow process whose stages can be defined and information flow between these stages partially automated.
- Traceability management
  - Automated retrieval of the links among requirements

Requirements change management

- Should apply to all proposed changes to the requirements
- Principal stages
  1. Problem analysis. Discuss requirements problems and propose changes
  2. Change analysis and costing. Assess effects and costs of change on other requirements
  3. Change implementation. Modify requirements document and other documents to reflect change

A common problem

If a requirements change to a system is urgently required, there is always a temptation to make that change to the system and then retrospectively modify the requirements document.

Resist this temptation!
Key points

- The process includes a feasibility study, and elicitation, analysis, specification, and management of requirements.
- Requirements analysis involves domain understanding, and requirements collection, classification, structuring, prioritization, and validation.
- Each system have multiple stakeholders with different requirements.

Key points (continued)

- Social and organizational factors influence system requirements.
- Requirements validation is concerned with checks for validity, consistency, completeness, realism and verifiability.
- Business changes inevitably lead to changes in requirements.
- Requirements management includes planning and change management.