1 Requirements Engineering

In the following we are gonna look at requirements engineering using the case study of a car sharing service. The main roles and functionalities of a car sharing service are:

> **Main Roles & Functionalities**
>
> - Role-Independent
>   - Authentication
> - Administrator
>   - Add / change new cars, rental locations
>   - Biling
> - User
>   - Check availability
>   - Request booking
>   - Change booking
> - Service Staff
>   - Take out vehicle for service

Requirements Analysis is concerned with building a system of what the product *needs* to fulfill in terms of budget, time and surrounding criteria.

So the objectives are akin to:

> **What has to be developed?**
>
> - Need to understand the problems that arise in the requirement elicitation phase
> - The different kinds of requirements
> - The requirement engineering workflow
> - Modelling requirements
>   - Scenarios & Use cases
>   - Notations: Textual and Graphical

Although the objectives seem pretty straightforward, requirements analysis can be tricky due to how ambigous language can be. Thorough communication is important to understand fully understand what the client wants.

> **What is Requirements Engineering?**
>
> The process of
> - finding
> - analysing
> - documenting
> - validating
>
> software requirements.

1.1 What are Requirements?

> **Definition**
>
> - Requirements as descriptions of the services provided by the system
>   - Car booking
>   - Service booking
>   - Location tracking
>   - etc.
> - Requirements as the operational constraints of the system
>   - Database throughput
>   - System memory
>   - Navigation systems
>   - etc.

These requirements are usually handled in the form of **System Requirements Specification (SRS)** Documents (Ger: Pflichtenheft) or **User stories**, structured natural language of use cases, state diagrams etc. stored in the product backlog (ordered list of requirements)

1.2 Different Types of Requirements

Overall the requirements can be divided in the following:

> - User Requirements
> - System Requirements
> - Functional Requirements
> - Non-Functional Requirements
> - Domain Requirements

1.2.1 User Requirements

> State in language or diagrams:
> - What services the system should provide
> - What the operational constraints are
>
> The descriptions are often high-level and abstract.

For example "According to german law, a car sharing service must keep track of all bookings"

1.2.2 System Requirements

> Precise and detailed specification of the systems
> - functions
> - services
> - operational constraints

For example: "After a successful booking the user must be shown an overview of their booking"

"Booking details must be stored for 10 years"

> **Characteristics**
> - Refinement of user Requirements
> - Determine system interface (functional)
> - Recorded as part of the SRS and part of the contract with the client
> - Authored by software developer or business analyst with the client

1.2.3 Functional Requirements

> Functionality that is clearly identifiable and localized in the code
> - Services providede by the system
> - System reactions to inputs or events
> - System behaviour in specific situations like Network distruption

1.2.4 Non-Functional Requirements (NFR)

> Constraints of the services or functions
> - Service Level Agreement (SLA)
> - Constraints from development process
> - Alignment to standards (e.g. Protocols)

NFRs often apply to the whole system as they cannot be handled by simply adding a peice of code.

For example: "The database must be able to process 1000 queries a second" "User data must only be accessible to authorized persons"

> **Examples of Non-Functional Requirements**
> - Product requirements
>   - Reliability (crashes, use cases)
>   - Efficiency (performance, memory)
>   - Portability (Not confined to one device or service)
> - Organisational Requirements
>   - Delivery mode (beta, continous)
>   - Implementation (Programming language, framework)
>   - Standardization (ISO standards or similar)
> - External requirements
>   - Interoperability (TUCaN ⇔ Moodle)
>   - Ethical aspects
>   - Legal aspects (safety, security, privacy)

NFRs may often result in the identification of functional requirements and are often more important to adhere to strictly than individual functional requirements.

A problem with NFRs come from how subjective they are: What is ethical, what is ease of use, what is good performance etc.?

1.2.5 Domain Requirements

Are derived from the application domain rather than the needs of the user
- Often expressed in domain specific language → Hard to understand for software engineers.
- For example: Software engineers usually do not have profound knowledge of chemistry, however the client might be a chemist and needs software that can be used for very specific applications.
- Often implicitly assumed as obvious to domain experts
- Can be functional or non-functional

1.3 Feasibility Study

The objective of the Feasibility Study is to obtain a justified understanding of whether the requirements engineering and system development phases should be **started**. This is usually base on:

- Business requirements
- Outline description of the system
- Description of how the system should support the business

The resulting **Feasibility Report** then covers

- Whether the system contributes to the objective of the organization
- Whether the system can be implemented within technical, financial and schedule constraints
- Whether the system can be implemented using other systems used by the company

1.4 Requirements Elicitation and Analysis

1.4.1 Requirements Discovery

Systematic Requirement Discovery
Viewpoint-Oriented Approach

- Interactor Viewpoint
  - People or systems who interact directly with the system
  - End Users, Administrators, Service Personal, etc.
  - **Direct Stakeholders**
- Indirect Viewpoint
  - Stakeholders who influence the requirements, but won't use the system directly
  - CFO, Data protection personal, etc.
  - **Indirect Stakeholders**
- Domain Viewpoint
  - Domain characteristics & constraints that influence the requirements
  - Legal, Ethical, etc.

The goal of the requirement elicitation process is to develop more specific viewpoints and use them to discover more specific requirements.

The elicitation can be done in an interview which are usually structured as follows:

> **Systematic Requirement Discovery**
> **Interviews**
>
> - Closed Interviews:
>   - Predefined questions
> - Open Interviews:
>   - No predefined agenda
> - Interviews should only be used as a supplement:
>   - Interviewee can be biased
>   - Interviewee can assume domain knowledge

Some further elicitation techniques are:

> **Systematic Requirement Discovery**
> **Other Techniques**
>
> - Scenario Analysis
>   - Analyses the sequence of interactions with the system
> - Use Case Analysis
>   - Analyses the use cases of the system

1.4.2 Requirements Classification & Organisation

For further structured workflow the requirements should be categorized, this can be done using the **FURPS+** Model:

> - **F**unction
> - **U**se
> - **R**equirements
> - **P**riority
> - **S**cope
> - **+**
>   - Implementation
>   - interface
>   - Operations
>   - Packaging
>   - Legal

1.4.3 Requirements prioritisation & Negotiation

Another problem in the elicitation process are conflicts. Different stakeholders might have different requirements. These conflicts need to be resolves through negotiation.

1.4.4 Requirements Documentation

The produced requirements are then documented and used as a basis for further elicitation and analysis. These documents (SRS) can be formal or informal.

SRS Target Groups

- Client, users
- Managers: Client and Manufacturer
- System Engineers, system testers, system maintainers
- Anyone concerned with ordering, using, manufacturing or maintaining

The level of detail of the SRS depends on the system, development process, whether the product is developed in-house or external etc.

The usual format of an SRS is:

System Requirement Specification (SRS) Document Format

1. Introduction
   (a) Purpose of the SRS
   (b) Scope of the product (Also what isn't in the scope)
   (c) Glossary
   (d) References
   (e) Overview
2. General Description
   (a) Product perspective
   (b) Product functions
   (c) User characteristics
   (d) Limitations
   (e) Assumptions and dependencies
3. Specific Requirements
4. Appendices, Index, etc.

1.4.5 Requirements Validation

Requirement Validation Checklist

- Validity
  - Do the requirements capture the needed features?
  - Is additional functionality needed?
- Consistency
  - Are the requirments conflicting?
- Completeness
  - Do the requirements cover all the features and constraints?
- Realism
  - Can the requirements be implemented feasably?
- Verifiability
  - Is there criteria to check whether the requirements are met?
- Traceability
  - Is each requirement tracable to the source of the requirement?