Software Requirements

Sommerville, Chapter 6

What are Software Requirements?

Requirements are capabilities and conditions to which a system (project) must conform.

Identifying, documenting and communicating requirements is a challenging task in the software development process

Requirements Identification

Often referred to as:

- Requirements gathering
- Requirements elicitation

Requirements Elicitation Techniques

Techniques used:

- Use case writing
- Workshops
- Questionnaires/surveys
- Informal Meetings
- Review of similar systems

The Importance of Well-defined Requirements

Misunderstood requirements results in the delivery of a system that does not meet the customers expectations.

The delivered system may be missing some essential functions and may have an abundance of non-essential requirements.

Validation and verification should minimize this risk.

Requirements ***management*** is different depending on the process model used:

- **Waterfall**: Requirements must be specified before design and implementation can commence
- **Agile**: embraces changing requirements during design and implementation

Objective of Requirements:

- To describe the services (functionality) provided by the system
- To describe the operational constraints of the system
- To reflect the needs of the customer
  - Controlling a device
  - Placing an order
  - Finding information

Requirements engineering (RE) is the process of finding out, analysing, documenting and validating these services and constraints.

Defining requirements can be different at different levels of the RE process. For this reason we distinguish between ***user*** requirements and ***system*** requirements.

User requirements: high-level abstract requirements

System requirements: a detailed description of what the system must do.

User Requirements

- Statements (simple)
- Represented by natural language and/or diagrams
- Describe the services (functions) the system **must** provide
- Describe operating constraints
- Written for the customer/client

System Requirements

A ***detailed*** description of the system services (functions) and operational constraints

This functional specification must be precise.

It ***may*** be part of the legal contract between the developer and the client

A single user requirement may be expanded into ***several*** system requirements.

– A user requirement is abstract

– A system requirements adds detail giving a detailed explanation of the services (functions) that must be provided.

Consider the following example:

A User requirement specification

StockSYS must allow details of each Supplier to be recorded

Corresponding system requirement specification

When a supplier registers with the business

- The supplier details are captured via the user interface from a completed application form
- The data entered is validated
  - *Describe validation checks to be made*
  - *Describe any business rules to be enforced*
- The system determines the next SupplierId to be assigned
- The SupplierId and validated supplier details are recorded in the ***Supplier File***

Is this system requirement written accurately?

Can it be improved in any way?

There may be a business rule(s) to be considered by the requirement.

The system requirement specification needs to reflect this.

Requirements Specification – Target Audience

User requirements readers are:

- End users/operators
- Managers

These readers are **NOT** concerned with:

- Implementation (development process)
- The detailed facilities of the system

System requirements readers are:

- Business analysts (BA)
- Software programmers
- QA (testers)
- IT Support
- Database Administrator
- Managers

These readers are concerned with:

- The business process (BA)
- System implementation (programs; database;operations)

Requirements Classification

Requirements are classified as:

- Functional requirements (services)
- Non-functional requirements (performance, etc)
- Domain requirements

Functional Requirements

- Statements of services to be provided
- System response to particular inputs
- System response to particular situations
- May explicitly state what the system should ***not*** do

Non-functional Requirements

- Constraints on services to be provided
  - Timing constraints
  - Development process constraints
  - Development standards constraints
- Apply to the system as a whole (generally)
- Often could be deciding factor on the survival of the system (e.g. reliability, cost, response time)

Domain Requirements

- Come from the application domain of the system
- They reflect the characteristics and requirements of that domain
- Can be functional or non-functional

  e.g. User interface requirements

Project Constraints

One or more constraints might be applied to a project:

- Budget
- Schedule (deadline)
- Resources (people!)
- Technology/Tools
- Skills

Any of these constraints may impact on the ***scope*** of the system (what can be included, what must be excluded).

Requirements may need to be ***prioritized*** (some agreed scale) for inclusion in one or more releases of the system.

Requirements may need to be ***excluded*** due to budget constraints.

The requirements analysis activity will almost certainly lead to the production of a ***statement of scope and objectives*** for the proposed system.

Distinctions between the types of requirements is not always clear-cut. Functional or non-functional?

A user requirement concerned with security may appear to be a non-functional requirement.

When developed in more detail, the requirement may generate additional requirements that are functional e.g. user authentication facilities (software)

Functional Requirements

- Describe what system should do
- Depend on type of software being developed
- Depend on the expected users of the software
- Depend on the organisation's approach to writing requirements

Functional user requirements are written in an abstract way.

Functional system requirements much more detailed.

Functional user requirements

1. The user shall be able to search the stock inventory for a required stock item
2. The system shall restrict access to data to different users
3. Every order shall be allocated a unique identifier (ORDER_ID) which the user can use to trace an order status.

Functional system requirements

The user may search the stock inventory for a particular stock item. This search may be performed by using the unique stock identifier (STOCK_ID), the description (or part of) of the stock item (STOCK_DESC) or the category of stock (STOCK_CAT) to which the stock belongs. All items matching the search criteria are retrieved from the information system (Stock File) and displayed on the user interface. The user may print the retrieved information if required.

Non-functional Requirements

- Relate to emergent system properties
  - Reliability
  - Response time
  - Storage requirements
  - Security
- Define system constraints (I/O Capabilities)
- Define data representation to be used in system interfaces

Non-functional requirements are often regarded as being more critical than functional requirements.

- Users can often work around a system function that doesn't meet their needs.
- Failing to meet a non-functional requirement ***may*** mean that the entire system is unusable.

  e.g reliability requirements of an aircraft system

Some non-functional requirements may impose constraints on the process used to develop the system:

- Specification of quality standards
- Design must be produced with a particular CASE tool

Sources of non-functional requirements:

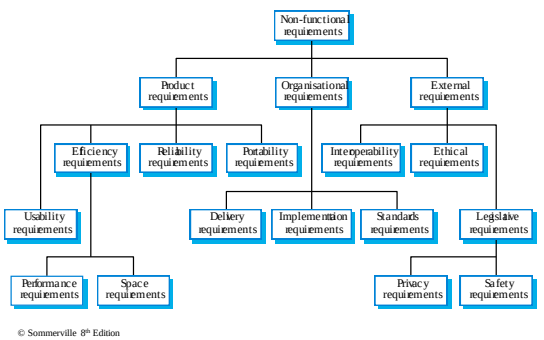
- User needs
- Budget constraints
- Organisational policies
- Need for interoperability with other systems
- Safety regulations
- Privacy legislation

FYI:

The 'IEEE-Std 830 - 1993' lists 13 non-functional requirements to be included in a Software Requirements Document.

- Performance requirements
- Interface requirements
- Operational requirements
- Resource requirements
- Verification requirements
- Acceptance requirements

- Documentation requirements
- Security requirements
- Portability requirements
- Quality requirements
- Reliability requirements
- Maintainability requirements
- Safety requirements



© Sommerville 8th Edition

Examples of non-functional requirements for StockSYS

Product Requirement

4.1 The user interface for StockSYS will be implemented as simple HTML without frames or Java applets.

Organisational Requirement

6.3.1 The system development process and deliverable documents shall conform to the process and deliverables defined in ISO 2001-98

External Requirement

10.2 The system shall not disclose any personal information about Suppliers to any staff other than the SupplierID and their name.

Non-functional requirements are sometimes difficult to verify. Some are stated as general goals:

- Ease of use
- Recovery from failure
- Rapid user response

An acceptable measure of these goals are often disputed on system delivery

Domain Requirements

Derived from the application domain of the system as opposed to the specific needs of the users.

DVDSYS may stipulate:

1. There shall be a standard user interface to all databases based on the ISO-2001-77 standard

2. All member registration forms must be destroyed following registration on the system.