

# Chapter 4

## Chapter 4 Requirements and Specification

### Learning Objective

... Establishing what the *customer* requires from a software system.

**Frederick T Sheldon**  
Assistant Professor of Computer Science  
Washington State University

---

---

---

---

---

---

---

---

## Objectives

- To introduce the notion of requirements engineering
- To explain *why requirements at different levels of detail are needed*
- To describe how the system requirements document may be organized
- To describe the *requirements validation process*
- To explain *why requirements evolve* during the lifetime of a system

---

---

---

---

---

---

---

---

## Topics covered

- The requirements engineering process
- The software requirements document
- Requirements validation
- Requirements evolution

---

---

---

---

---

---

---

---

# Requirements engineering

The process of establishing the services that the customer requires from a system and the constraints under which it operates and is developed

Requirements may be functional or non-functional

- Functional requirements describe system services or functions
- Non-functional requirements is a constraint on the system or on the development process

---

---

---

---

---

---

---

---

# What is a requirement?

It may range from a *high-level abstract statement* of a service or of a system constraint to a *detailed mathematical functional specification*

This is inevitable as requirements may serve a *dual* function

- May be the basis for a *bid for a contract*- therefore must be open to interpretation
- May be the basis for the *contract itself*- therefore must be defined in detail
- Both these statements may be called requirements*

---

---

---

---

---

---

---

---

# Requirements definition/specification

## Requirements definition

A statement in *natural language plus diagrams* of the services the system provides and its operational constraints. Written for customers

## Requirements specification

A *structured document setting out detailed descriptions* of the system services. Written as a *contract* between client and contractor

## Software specification

A *detailed software description which can serve as a basis for a design or implementation*. Written for developers

---

---

---

---

---

---

---

---

# Definitions and specifications

## Requirements definition

1. The software must provide a means of representing and accessing external files created by other tools.

## Requirements specification

- 1.1 The user should be provided with facilities to define the type of external files.
- 1.2 Each external file type may have an associated tool which may be applied to the file.
- 1.3 Each external file type may be represented as a specific icon on the user's display.
- 1.4 Facilities should be provided for the icon representing an external file type to be defined by the user.
- 1.5 When a user selects an icon representing an external file, the effect of that selection is to apply the tool associated with the type of the external file to the file represented by the selected icon.

---

---

---

---

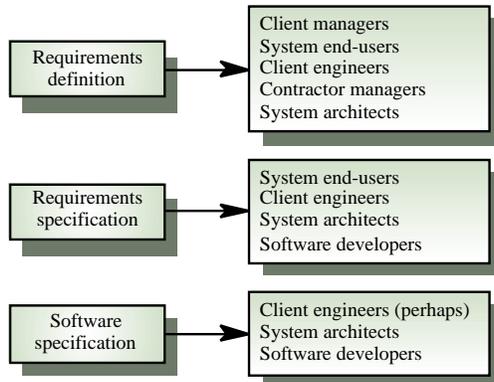
---

---

---

---

# Requirements readers



---

---

---

---

---

---

---

---

# Wicked problems

Most large software systems address *wicked problems*

*Problems which are so complex that they can never be fully understood* and where understanding develops during the system development

Therefore, *requirements are normally both incomplete and inconsistent*

---

---

---

---

---

---

---

---



## The requirements document

The requirements document is the *official statement of what is required* of the system developers

Should include both a definition and a specification of requirements

It is **NOT** a design document. As far as possible, it should set of **WHAT** the system should do rather than **HOW** it should do it

---

---

---

---

---

---

---

---

## Requirements document *requirements*

*Specify external system behavior*

Specify implementation *constraints*

*Easy to change*

*Serve as reference tool* for maintenance

*Record forethought* about the life cycle of the system i.e. predict changes

*Characterize responses* to unexpected events

---

---

---

---

---

---

---

---

## Requirements document structure

### Introduction

Describe need for the system and how it fits with business objectives

### Glossary

Define technical terms used

### System models

Define models showing system components and relationships

### Functional requirements definition

Describe the services to be provided

---

---

---

---

---

---

---

---

## Requirements document structure

### Non-functional requirements definition

Define constraints on the system and the development process

### System evolution

Define fundamental assumptions on which the system is based and anticipated changes

### Requirements specification

Detailed specification of functional requirements

### Appendices

System hardware platform description

Database requirements (as an ER model perhaps)

### Index

---

---

---

---

---

---

---

---

## 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

Prototyping is an important technique of requirements validation

Discussed in Chapter 8

---

---

---

---

---

---

---

---

## 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

---

---

---

---

---

---

---

---

## 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*. Good communications between developers, customers and users can resolve problems at an early stage

---

---

---

---

---

---

---

---

## 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?

---

---

---

---

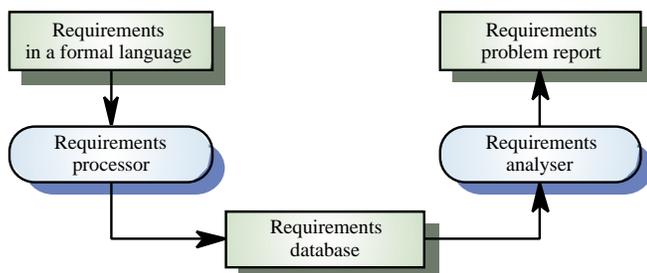
---

---

---

---

## Automated consistency checking



---

---

---

---

---

---

---

---

## Requirements evolution

*Requirements always evolve* as a better understanding of user needs is developed and as the organization's objectives change  
It is essential to *plan for change* in the requirements as the system is being developed and used

---

---

---

---

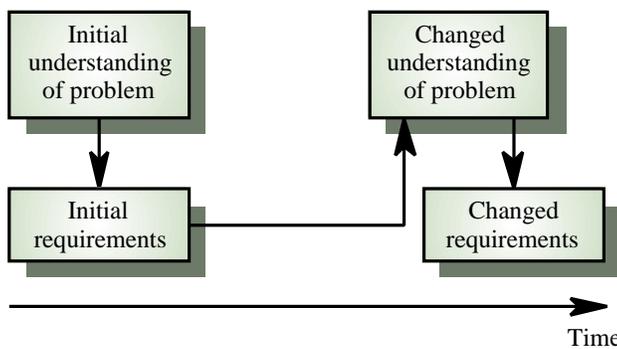
---

---

---

---

## Requirements evolution



---

---

---

---

---

---

---

---

## Requirements classes

*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 document changes

The requirements document should be organized so that *requirements changes can be made without extensive rewriting*

*External references should be minimized* and the document sections should be as *modular* as possible

Changes are easiest when the document is *electronic*. Lack of standards for electronic documents make this difficult

---

---

---

---

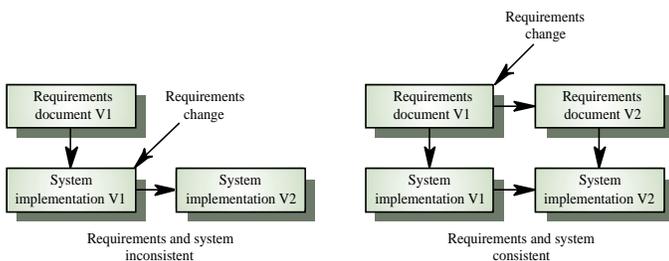
---

---

---

---

# Controlled evolution



---

---

---

---

---

---

---

---

## Key points

---

It is very difficult to formulate a complete and consistent requirements specification

A requirements definition, a requirements specification and a software specification are ways of specifying software for different types of reader

The requirements document is a description for customers and developers

---

---

---

---

---

---

---

---

## Key points

---

Requirements errors are usually very expensive to correct after system delivery

Reviews involving client and contractor staff are used to validate the system requirements

Stable requirements are related to core activities of the customer for the software

Volatile requirements are dependent on the context of use of the system

---

---

---

---

---

---

---

---