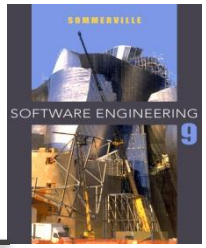
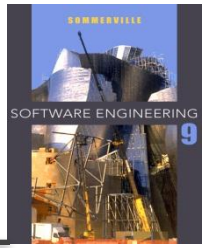


Chapter 4 – Requirements Engineering

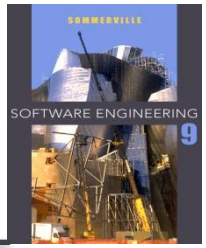
Lecture 1

Topics covered



- Functional and non-functional requirements
- The software requirements document
- Requirements specification
- Requirements engineering processes
- Requirements elicitation and analysis
- Requirements validation
- Requirements management

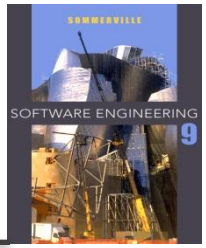
Requirements Engineering



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

- The requirements themselves are the **descriptions of the system services and constraints** that are generated during the requirements engineering process.

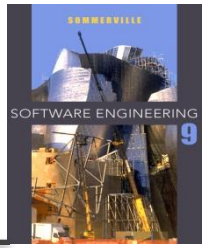
What is a Requirement?



- 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 Abstraction (Davis)



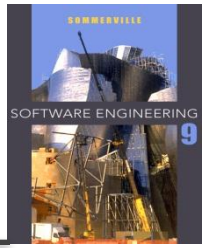
"If a company wishes to let a contract for a large software development project, it must define its needs in a sufficiently abstract way that a solution is not pre-defined.

The requirements must be written so that several contractors can bid for the contract, offering, perhaps, different ways of meeting the client organization's needs.

Once a contract has been awarded, the contractor must write a system definition for the client in more detail so that the client understands and can validate what the software will do.

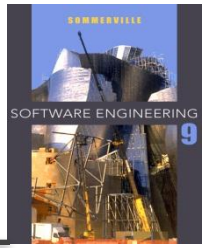
Both of these documents may be called the requirements document for the system."

Types of Requirements



- User requirements
  - Statements in natural language plus diagrams of the services the system provides and its operational constraints.
  - **Written for customers.**
- System requirements
  - A **structured** document setting out **detailed descriptions** of the system's **functions**, **services** and **operational constraints**.
  - Defines what should be implemented so may be part of a contract between client and contractor.
  - **Whom do you think these are written for?**
  - **These are higher level than functional and non-functional requirements, which these may subsume.**

User and System Requirements



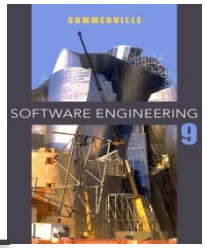
User requirement definition

1. The MHC-PMS shall generate monthly management reports showing the cost of drugs prescribed by each clinic during that month.

System requirements specification

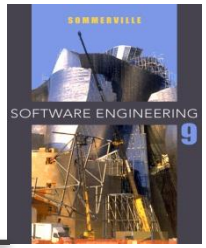
1.1 On the last working day of each month, a summary of the drugs prescribed, their cost and the prescribing clinics shall be generated.
1.2 The system shall automatically generate the report for printing after 17.30 on the last working day of the month.
1.3 A report shall be created for each clinic and shall list the individual drug names, the total number of prescriptions, the number of doses prescribed and the total cost of the prescribed drugs.
1.4 If drugs are available in different dose units (e.g. 10mg, 20 mg, etc.) separate reports shall be created for each dose unit.
1.5 Access to all cost reports shall be restricted to authorized users listed on a management access control list.

Functional and Non-functional requirements



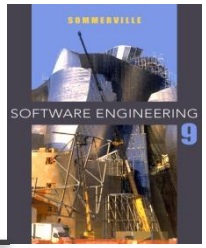
- Functional requirements
  - Statements of **services** the system should provide, how the system should react to particular inputs and how the system should behave in particular situations.
  - May state what the system **should not do**.
- Non-functional requirements
  - **Constraints** on the services or functions offered by the system such as timing constraints, constraints on the development process, standards, etc.
  - Often apply to the **system as a whole** rather than individual features or services.
- Domain requirements
  - Constraints on the system from the domain of operation

Functional Requirements



- Describe functionality or system services.
- Depend on the type of software, expected users and the type of system where the software is used.
- Functional user requirements may be high-level statements of what the system should do.
- Functional system requirements should describe the system services in detail.
- Essentially, these are the ‘whats’ of the system that we often refer to. These are not ‘all that there is,’ but these should describe the overall functionality of the system.

Non-functional Requirements

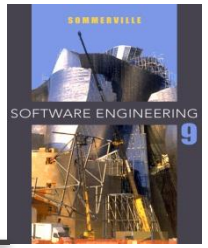


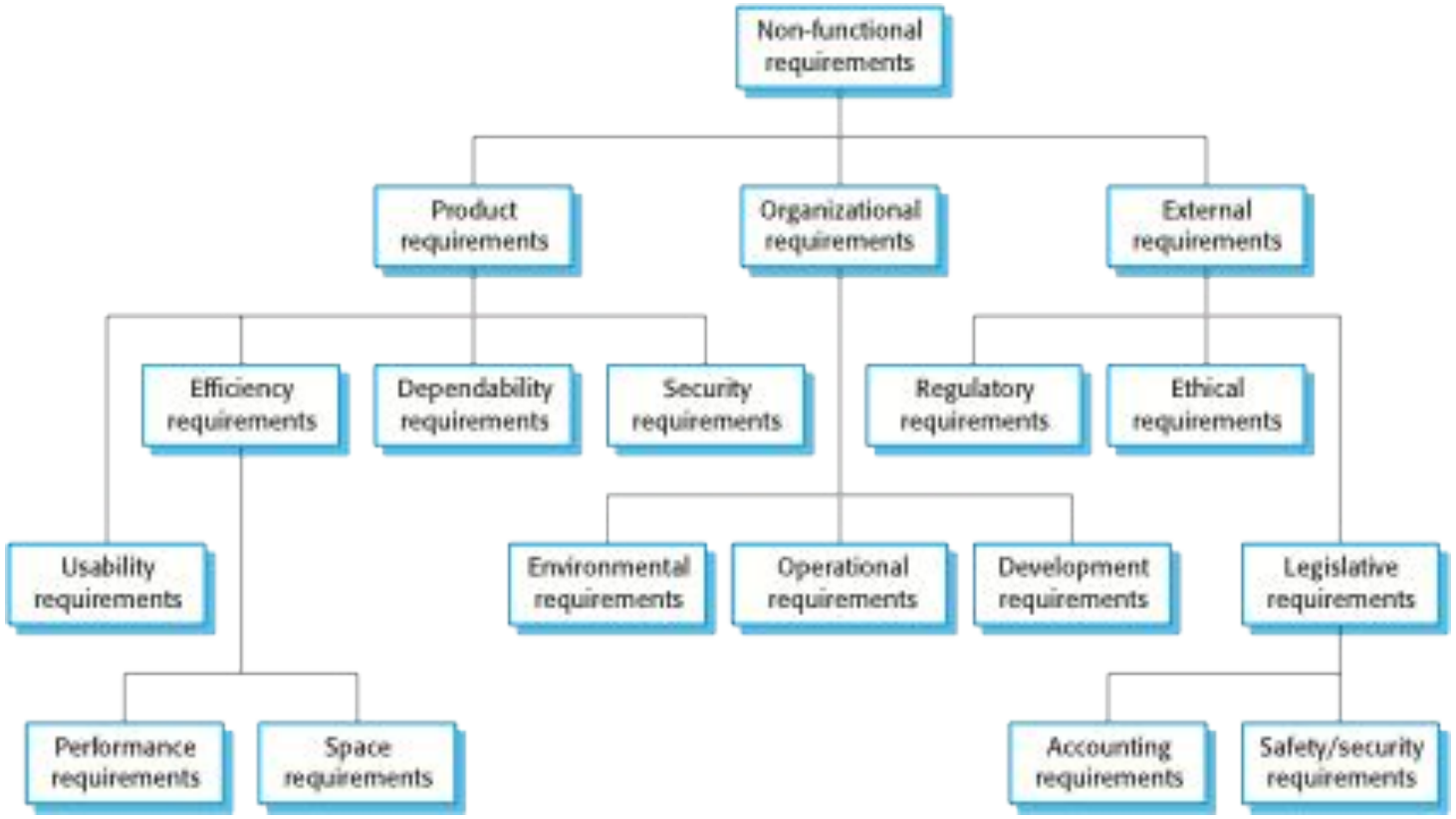
- These define system properties and constraints e.g. reliability, response time, maintainability, scalability, portability, and storage requirements.
- Constraints are I/O device capability, system representations, etc.

- Process requirements may also be specified mandating a particular IDE, programming language or development method.
- (Often internal to an organization or required for fit / compatibility with other comparable systems.)

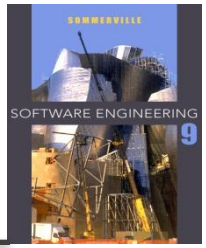
- Non-functional requirements **may be more critical** than functional requirements. If these are not met, the system may be useless.

Types of Nonfunctional Requirements



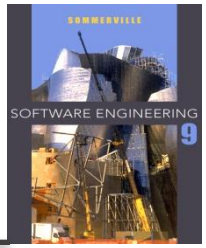


Non-functional Requirements Implementation



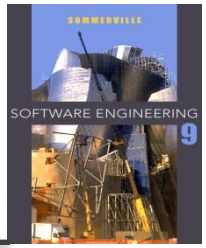
- Non-functional requirements may affect the **overall architecture of a system** rather than the individual components.
  - For example, to ensure that performance requirements are met, you may have to organize the system to minimize communications between components.
- A single non-functional requirement, such as a security requirement, may generate a **number** of related functional requirements that define system services that are required.
  - It may also generate requirements that **restrict** existing requirements.

Metrics for specifying nonfunctional requirements



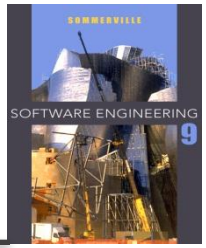
Property	Measure
Speed	Processed transactions/second User/event response time Screen refresh time
Size	Mbytes Number of ROM chips
Ease of use	Training time Number of help frames
Reliability	**Mean time to failure (MTTF)** Probability of unavailability Rate of failure occurrence Availability
Robustness	Time to restart after failure **(MTTR)** Percentage of events causing failure Probability of data corruption on failure
Portability	Percentage of target dependent statements Number of target systems

Domain requirements problems



- Understandability
  - Requirements are expressed in the language of the application domain;
    - Application written for mortgage banking people need to express functionality in terms of home loans, mortgage balances, escrow, investor accounting, foreclosure, etc.
  - This is **often** not understood by software engineers developing the system.
- Implicitness
  - Domain specialists understand the area so well that they do not think of making the domain requirements explicit.
  - **And this is often a major problem in communications!!!**

Key points



- **Requirements** for a software system set out what the system should do and define constraints on its operation and implementation.
- **Functional requirements** are statements of the **services** that the system must provide or are **descriptions of how some computations** must be carried out.
- **Non-functional requirements** often **constrain** the system being developed and the development process being used.
- They often relate to the emergent properties of the system and therefore apply to the system as a whole.