


Introduction to Requirements

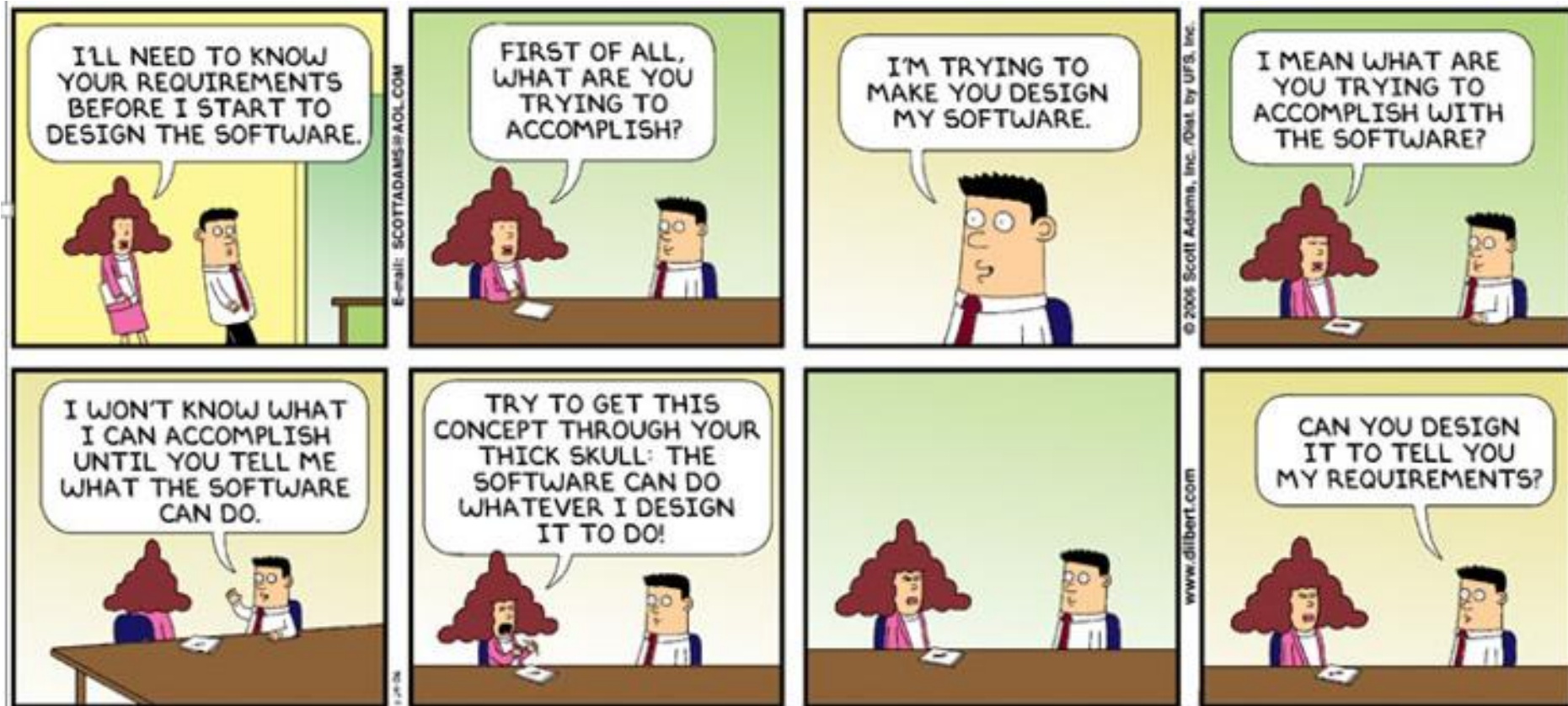
Course Topics

- ~~Why Requirements Engineering?~~
- Introduction to Requirements
- Requirements Engineering Methods
- System Vision, Context, and RE Framework
- Fundamentals of Goal Orientation
- Fundamentals of Scenarios
- Requirements Discovery
- User Stories and Agile Estimation
- Features Prioritization
- Requirements Negotiation
- Requirements Validation
- Fundamentals of Requirements Management

Lecture Objectives



- ✓ The difference between functional requirements, quality requirements, and development constraints
 - ✓ The differentiation between the problem ("what?") and the solution ("how?") with regard to the different phases of the development process
 - ✓ The influence of constraints on requirements
- 



The Term Requirements (IEEE 610.12-1990 standard)



A condition or capability

- (1) ...needed by a user to solve a problem or achieve an objective.
- (2) ... that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed documents.
- (3) A documented representation of (1) or (2).

(1) **Needs and goals of users**


(2) **Conditions and properties of the system to be developed**

(3) **A documented requirement is called “Requirements Artefact”**



Example of Requirements for a House Information System




- **R1:** The system shall offer a user-friendly interface.
 - **R2:** The house information system shall generate a monthly report containing all granted and denied admittances to the house.
 - **R3:** If the PIN (personal identification number) that the user enters at the keypad is correct, the system shall open the door and record the granted access, i.e., it should record the date and time, and the name of the PIN owner.
 - **R4:** The system shall be available on the market by May 1st, 2019.
 - **R5:** The unlocking of the door shall happen within 0.8sec after the PIN has been entered correctly.
- 

Requirements Abstraction



“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.”



User vs. System Requirements

■ User requirements

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

■ System requirements

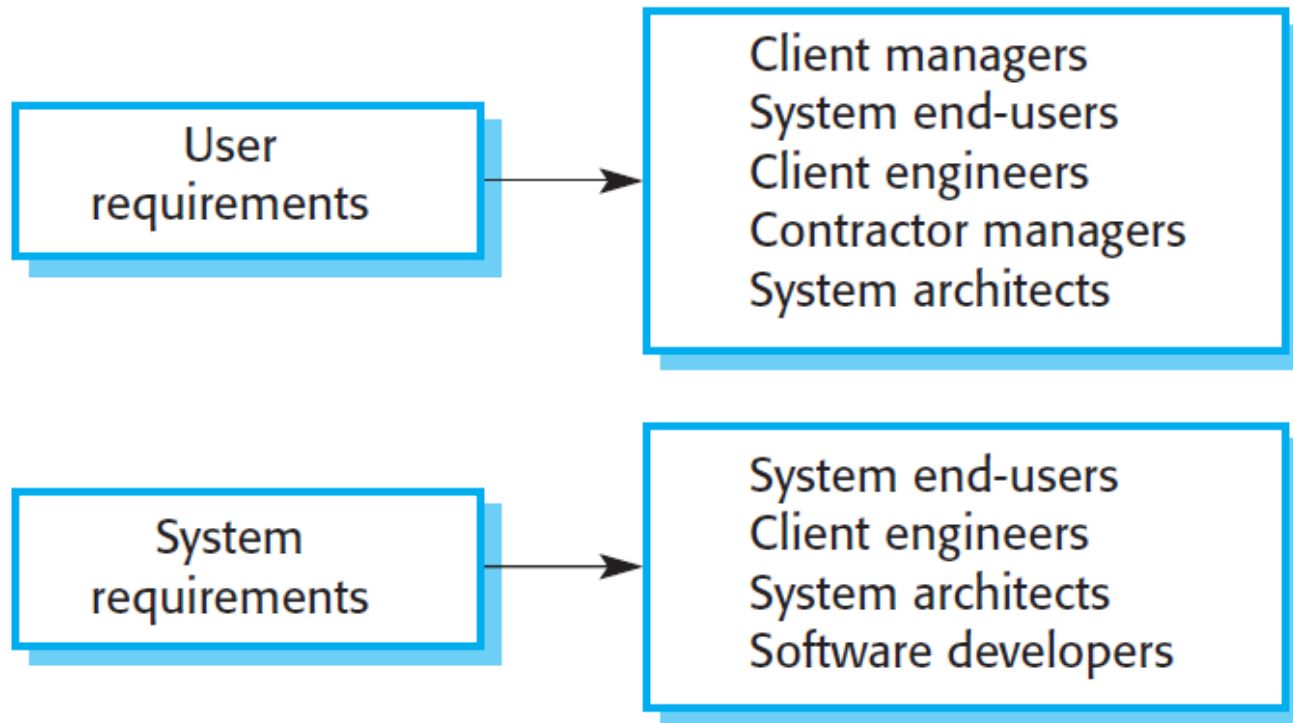
- A **structured document** setting out **detailed descriptions** of the system's functions, services and operational constraints.
- Defines what should be implemented so may be **part of a contract** between **client and contractor.**

■ Why?

- **Different levels of requirements** are useful because they communicate information about the system to **different types of readers.**

User vs. System Requirements: Different Types of Readers

A user requirement may be expanded into several system requirements.



User vs. System Requirements

A user requirement may be expanded into several system requirements.

User requirements definition

- 1.** The Mentcare system 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 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, 20mg, etc.) separate reports shall be created for each dose unit.
- 1.5** Access to drug cost reports shall be restricted to authorized users as listed on a management access control list.

In Class Activity 1



Consider the **university student registration system**:

From a **student perspective**:

- Write one **user requirement** for the system
 - Develop the corresponding **system requirement**
- 

Three Requirements Types



1. Functional requirements

Define services the system should provide, behavior of the system and in some cases also what the system should not do

2. Quality requirements

Define quality properties of the system, a component, a service or a function (e.g., performance)

3. Constraints

An organizational or technological requirement that restricts the way in which the system shall be developed



Functional Requirements



"These [**functional requirements**] are **statements of services** the system should provide, how the system should **react to particular inputs** and how the system should **behave in particular situations**. In some cases, the functional requirements may also **state what the system should not do**.

[...]

When expressed as **user requirements**, the requirements are usually described in a fairly abstract way. However, **functional system requirements** describe the **system function in detail, its inputs and outputs, exceptions, and so on**."

[Sommerville 2007]




Examples of Functional Requirements



A **functional requirement** defined for a security system of a building:

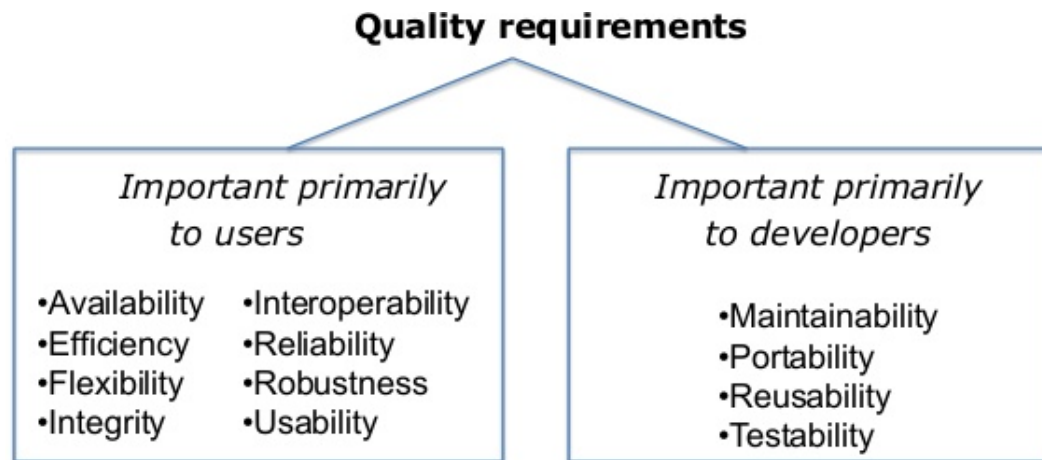
- **R1:** If a sensor detects that a glass pane is damaged or broken, the system shall inform the security company.

Other Examples:

- Deletion of an order will automatically delete all the lines of the order.
 - The image viewer must display enlarged images.
 - Logon shall require a valid User Id, User Password, and User Domain.
 - Three invalid logon attempts shall result in the current session being locked out for five minutes.
 - Six invalid logon attempts by a unique User Id shall result in the User Id being disabled.
- 

Quality Requirements


- A **quality requirement** defines a **quality property** of the entire system or of a system component, service, or function.
- Many taxonomies for quality requirements.



A quality requirements taxonomy, as defined in [Wiegiers 2003]

Quality Requirements (Important to users)



- **Availability:** refers to the percentage of time during which the system is actually available for use and fully operational.
 - **Efficiency:** is a measure of how well the system utilizes hardware resources such as processor time, memory, or communication bandwidth.
 - **Flexibility:** indicates how much effort is needed to extend the system with new capabilities.
 - **Integrity:** denotes how well the system is protected against unauthorized access, violations of data privacy, information loss, and infections through maleficent software.
 - **Interoperability:** indicates how easily the system can exchange data or services with other systems.
 - **Reliability:** is the probability of the system executing without failure for a specific period of time.
 - **Robustness:** is the degree to which a system or component continues to function correctly when confronted with invalid inputs, defects in connected systems or components, or unexpected operating conditions.
 - **Usability:** measures the effort the user requires to prepare input for, operate, and interpret the output of the system.
- 

Quality Requirements (Important to developers)



- **Maintainability**: indicates how easy it is to correct a defect or make a change in the system.
- **Portability**: relates to the effort it takes to migrate a system or component from one operating environment to another.
- **Reusability**: indicates the extent to which a component can be used in systems other than the one for which it was initially developed.
- **Testability** refers to the ease with which the software components or integrated system can be tested to find defects.

Examples of Quality requirements

- The system functions in a 7x24 mode and must have less than 1 hour downtime per month.
- The response time of the home page must not exceed five seconds.
- The system must be able to handle 1,000 concurrent web users per second.
- The parameter setting of the system can only be entered and modified by a user with super-user rights.

Quality requirements must have a **business case**, and are not for the sake of the development team.

Example: “To improve maintainability, the architecture of the Sales module must be refactored.”

Problem with Non-functional Requirements?

Term used in the literature

1. **Mainly Underspecified functional requirements:** Quite often documents an underspecified functional requirement. In such a case, we strongly recommend refining the underspecified functional requirement,
2. *Few of them are Quality requirements:* what remains after separating out the underspecified functional requirements

Non-functional requirements =

Underspecified functional
requirements


Quality requirements

Problem with Non-functional Requirements?



Example: R1: The system shall be secure.

Several questions arise due to the under specification of this requirement:

- What does the adjective "secure" mean?
 - Which properties shall the system provide in order to be "secure"?
 - How can one check whether the implemented system is "secure"?
- 

What About Non-functional Requirements?

Refinement of the underspecified requirement:

- **R1.1:** Each user must log in to the system with his user name and password prior to using the system. (*functional requirement*)
- **R1.2:** The system shall remind the user every four weeks to change the password. (*functional requirement*)
- **R1.3:** When the user changes his password, the system shall validate that the new password is at least eight characters long and contains alphanumeric characters. (*functional requirement*)
- **R1.4:** The user passwords stored in the system must be protected against password theft. (*quality requirement - integrity*)

Constraints

- A constraint is an organizational or technological requirement that restricts the way in which the system shall be developed.

[Robertson and Robertson 2006]

- **Types of constraints:** **cultural** (i.e. constraints originating from the cultural background of the system users), **legal constraints** (i.e. constraints originating from laws and standards), **organizational constraints**, **physical constraints**, **project constraints**, etc.
- Another categorization of constraints is to consider the subject affected by a constraint:
 - The system itself (e.g., it may demand or restrict a system function) or
 - The development process of the system (e.g., the process model to be used).


Example of Constraints



Constraints affecting the system:

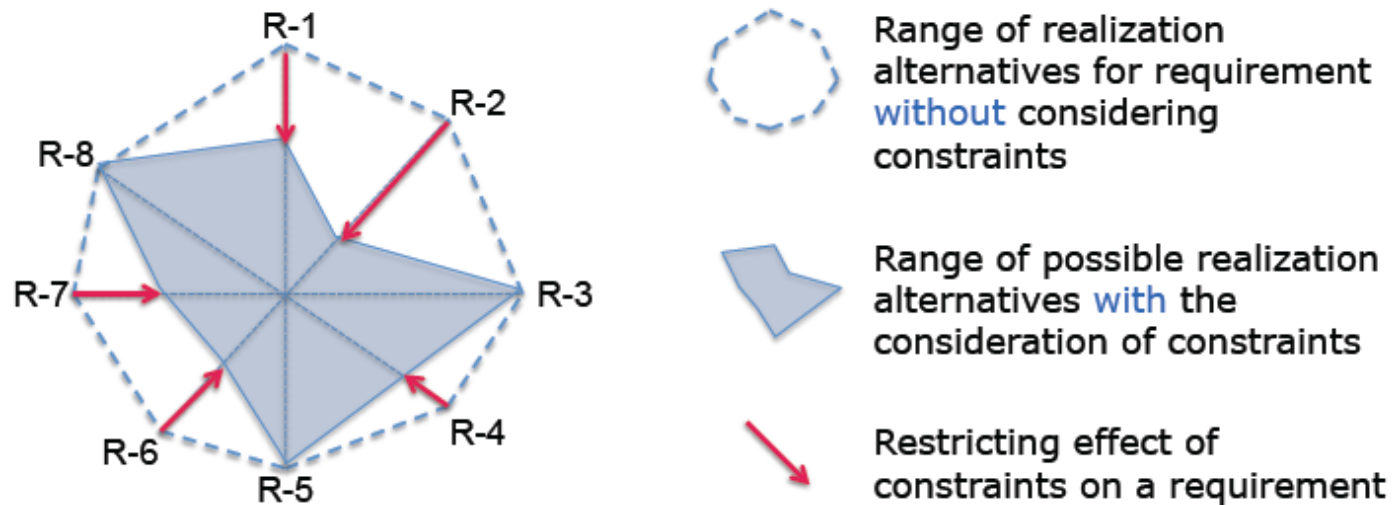
- **C1:** Due to current conditions defined by the insurance company, only the security technician is allowed to deactivate the control function of the system.
- **C2:** A fire protection requirement demands that the terminals in the sales rooms do not exceed the size 120 cm (height) x 90 cm (width) x 20 cm (depth).

Constraints affecting the development process:

- **C3:** The effort for the development of the system must not exceed 48 person/months.
 - **C4:** The system must be developed using the Rational Unified Process.
- 

Restrictions Imposed by Constraints

- Constraints restrict the range of alternatives available for realizing the requirements/the entire system.




Impact of Constraints



- Constraints may also lead to the change of requirements or the definition of new requirements

Example:

- Original requirement for the security system of a building based on a stakeholder interview:
 - **R-F-17:** A personal password shall be used for the authentication of each person at the access control system.
- 

Impact of Constraints

- During an interview, the director of a security firm mentions that according to a governmental law the access control for such buildings must comply with **the standard 77/12/EG**.
- On reading the standard, the requirements engineer recognizes that the **standard forbids the authentication with a personal password for such buildings**. Instead, the standard requires **authentication with a fingerprint sensor and with an accuracy of at least 60 minutiae**.

This newly identified constraint leads to **a change of the requirement R-F-17** as well as to the new requirement R-Q-4.

- **R-F-17**: A fingerprint sensor has to be used for the authentication of each person at the access control system.
- **R-Q-4**: The authentication must be performed with an accuracy of at least 60 minutiae.

Another example of a constraint effect

- Students were asked to develop a new website for Business Informatics. One of the requirements was defined as:
 - **R:** People without much knowledge of HTML and scripting should be able to maintain the website.
- The idea was to implement a content management system (CMS), such as Joomla, in order to satisfy the requirement above.
- In addition to the requirements, we defined the following constraint:
 - The website should run on the ICS department's servers.
- Soon we found out that the ICS servers do not support MySQL.
- Hence, a CMS using MySQL, such as Joomla, was no longer an option.

Examples

If the sensor detects that a glass pane is damaged or broken, the system shall inform the security company



Functional
requirement



Quality
requirement



Constraint

Examples

The house information system shall generate a monthly report containing all granted and denied admittances to the house



Functional
requirement



Quality
requirement



Constraint

Examples

The system must be developed using Java



Functional
requirement



Quality
requirement



Constraint

Examples

Valid logon transaction response shall occur within 1sec of the request when the system architecture is under nominal and peak transaction loads



Functional
requirement



Quality
requirement




Constraint

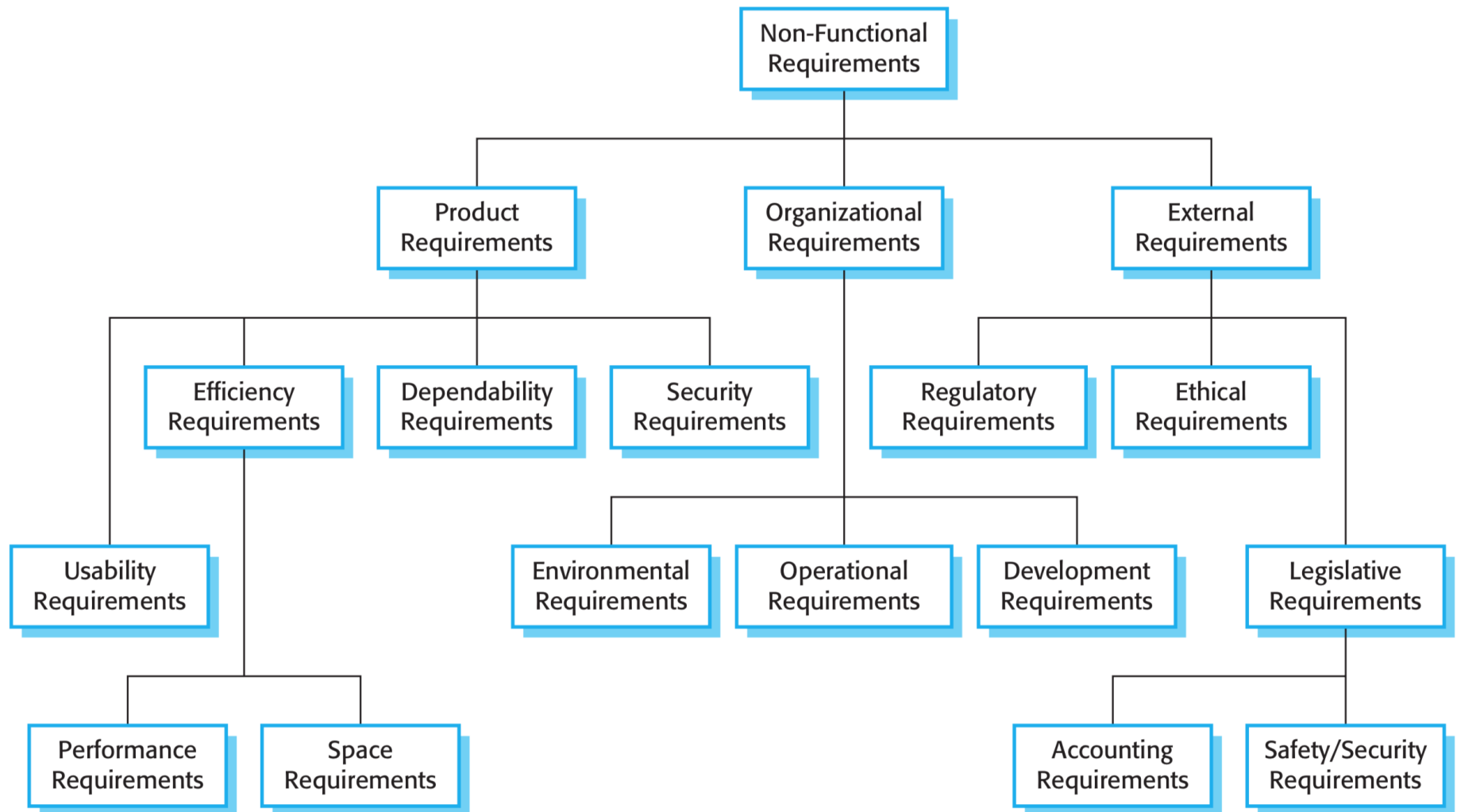
In Class Activity 2



Consider the **university student registration system**:
From a student perspective:

- Propose **1 functional requirement**
 - Propose **1 quality requirement.**
 - Propose **1 constraint.**
- 

Non-functional classifications




Non-functional classifications



- **Product requirements** - system must behave in a particular way
e.g.
 - execution speed, memory usage, usability requirements, reliability requirements (set out the acceptable failure rate), etc.

 - **Organizational requirements** – derived from organizational policies and procedures, e.g.
 - process standards used, implementation requirements, development environment, etc.

 - **External requirements** - factors which are external to the system e.g.
 - Regulatory requirements, interoperability requirements, legislative requirements, etc.
- 

Examples of Non-functional requirements

■ Product requirement

- The user interface for LIBSYS shall be implemented as simple HTML without frames or Java applets.
- The Mentcare system shall be available to all clinics during normal working hours (Mon–Fri, 08:30–17:30). Downtime within normal working hours shall not exceed 5 seconds in any one day.

■ Organisational requirement

- The system development process and deliverable documents shall conform to the process and deliverables defined in XYZCo-SP-STAN-95.
- Users of the Mentcare system shall identify themselves using their health authority identity card.

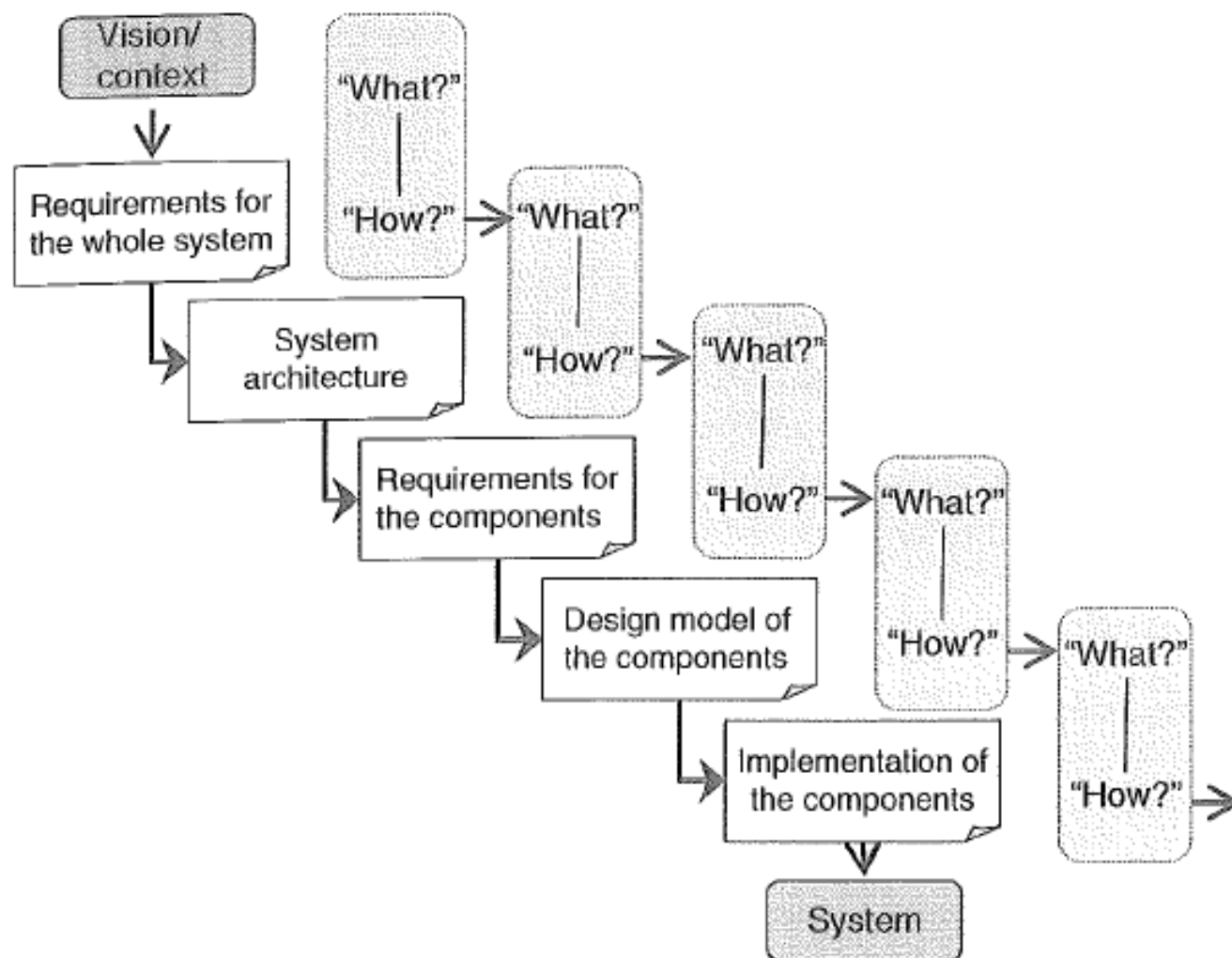
■ External requirement

- The hotel management system should comply with the Ministry of Tourism guidelines and standards as documented in ...
- The system shall implement patient privacy provisions as set out in HStan-03-2006-priv.

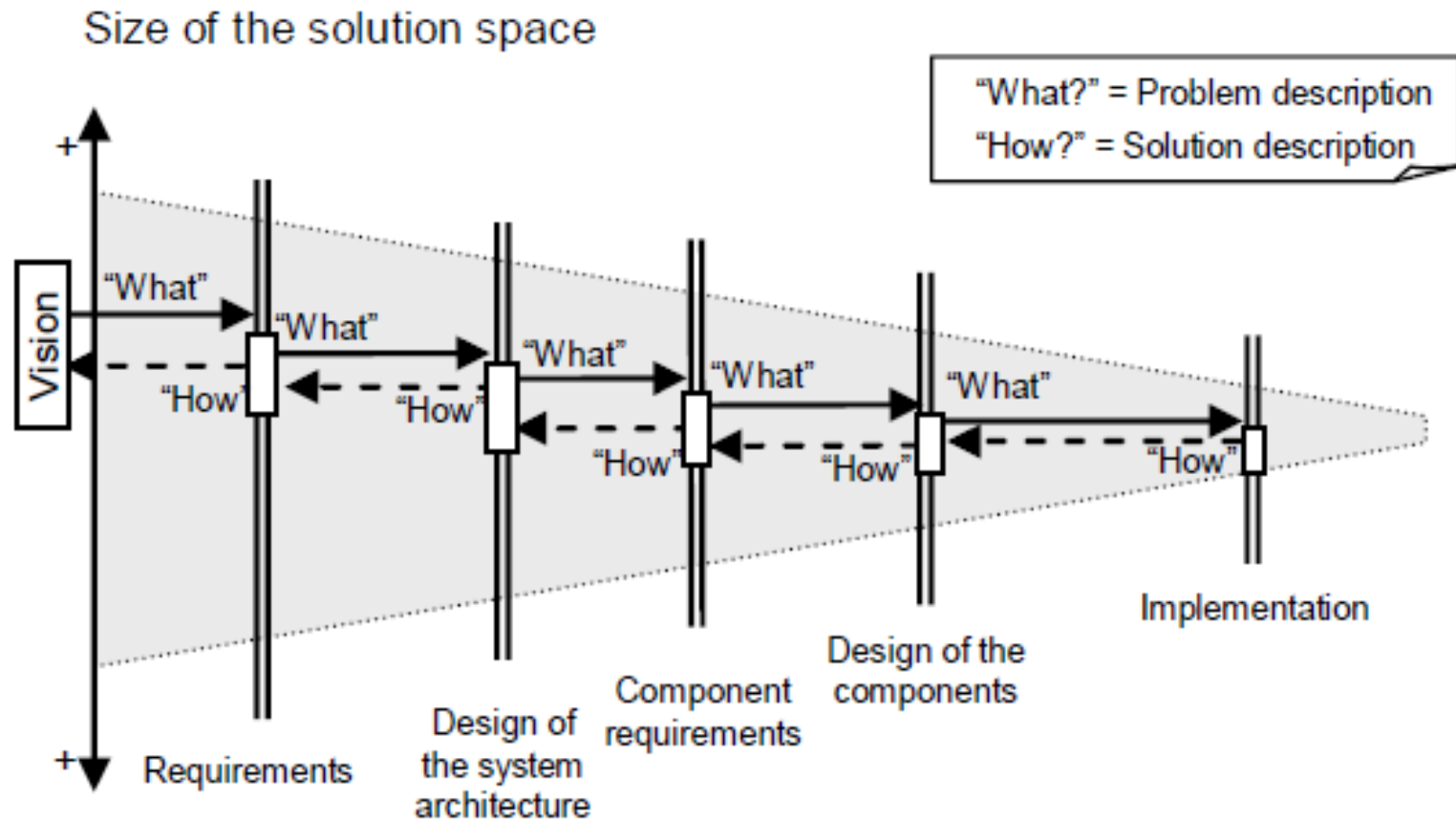
Problem vs. Solution

- “What?” vs. “How?”
 - Different views for different stakeholders, e.g.
 - **Requirements engineering:**
 - “What?” – system vision
 - “How?” – specified requirements
 - **System architect:**
 - “What?” – system requirements
 - “How?” – resulting system
- **Solution space decreases during the development process**
- **Interactions between requirements and design stage possible**

“What” vs “How” in the System Development Process



Reduction of the Solution Space During the Development Process



What? vs. How? During Requirements Engineering



- The differentiation between "what" and "how" also **occurs within a single development phase.**
- During requirements engineering a **high-level requirement** can define the "what" to be "realized" by a set of **more detailed requirements** which, in this case, define the "how" for realizing the more abstract requirement.


What? vs. How? During Requirements Engineering



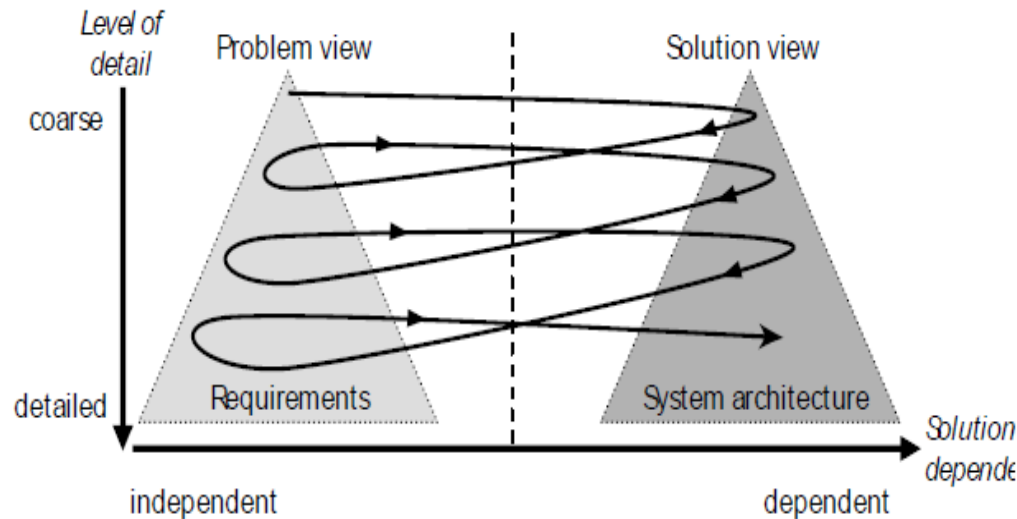
Problem definition - "What?"

- **R-1:** The navigation system shall allow the driver to enter the destination of the trip conveniently.

Solution description - "How?"

- **R1.1:** When the driver starts a new trip, the navigation system shall display a roadmap of the area centered on the Current position.
 - **R1.2:** The navigation system shall allow the driver to scroll and zoom into the roadmap.
 - **R1.3:** After the driver has selected a destination on the roadmap, the system shall allow the driver to edit the destination details (city, street, and house number).
- 

Interaction Between Requirements Engineering and Architectural Design



The twin-peaks model [Nuseibeh 2001]

The problem definition is typically not independent of the solution definition. A large proportion of requirements are specified with a (preliminary) solution in mind.