HUST

TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

ONE LOVE. ONE FUTURE.

SOICT

School of Information and Communication Technology

ONE LOVE. ONE FUTURE.

TRƯỜNG ĐẠI HỌC
BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY
OF SCIENCE AND TECHNOLOGY

IT3180 – Introduction to Software Engineering

7 – Requirement Analysis

ONE LOVE. ONE FUTURE.

Requirements

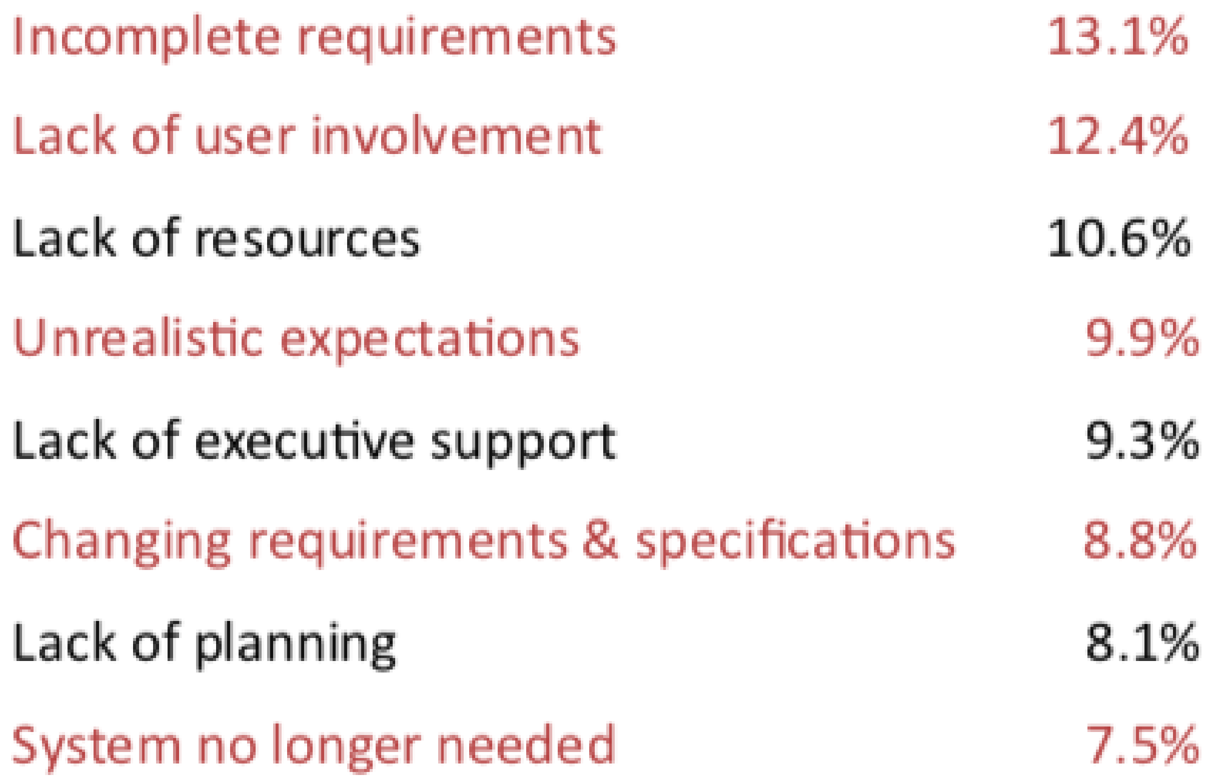
Requirements describe the function of the system from the client's viewpoint.

- The requirements establish the system's functionality, constraints, and goals.

- The requirements must be understandable by both the client and the development staff.
- SRS = Software Requirement Specification
- The development team and the client need to work together closely to define the requirements.

Why are Requirements Important?

Causes of failed software projects



Incomplete requirements	13.1%
Lack of user involvement	12.4%
Lack of resources	10.6%
Unrealistic expectations	9.9%
Lack of executive support	9.3%
Changing requirements & specifications	8.8%
Lack of planning	8.1%
System no longer needed	7.5%

Failures to understand the requirements led the developers to build the wrong system

Steps in Defining the Requirements

Defining the requirements can be divided into several steps:

- **Analysis** to establish the functionality by consultation with client, customers, and users

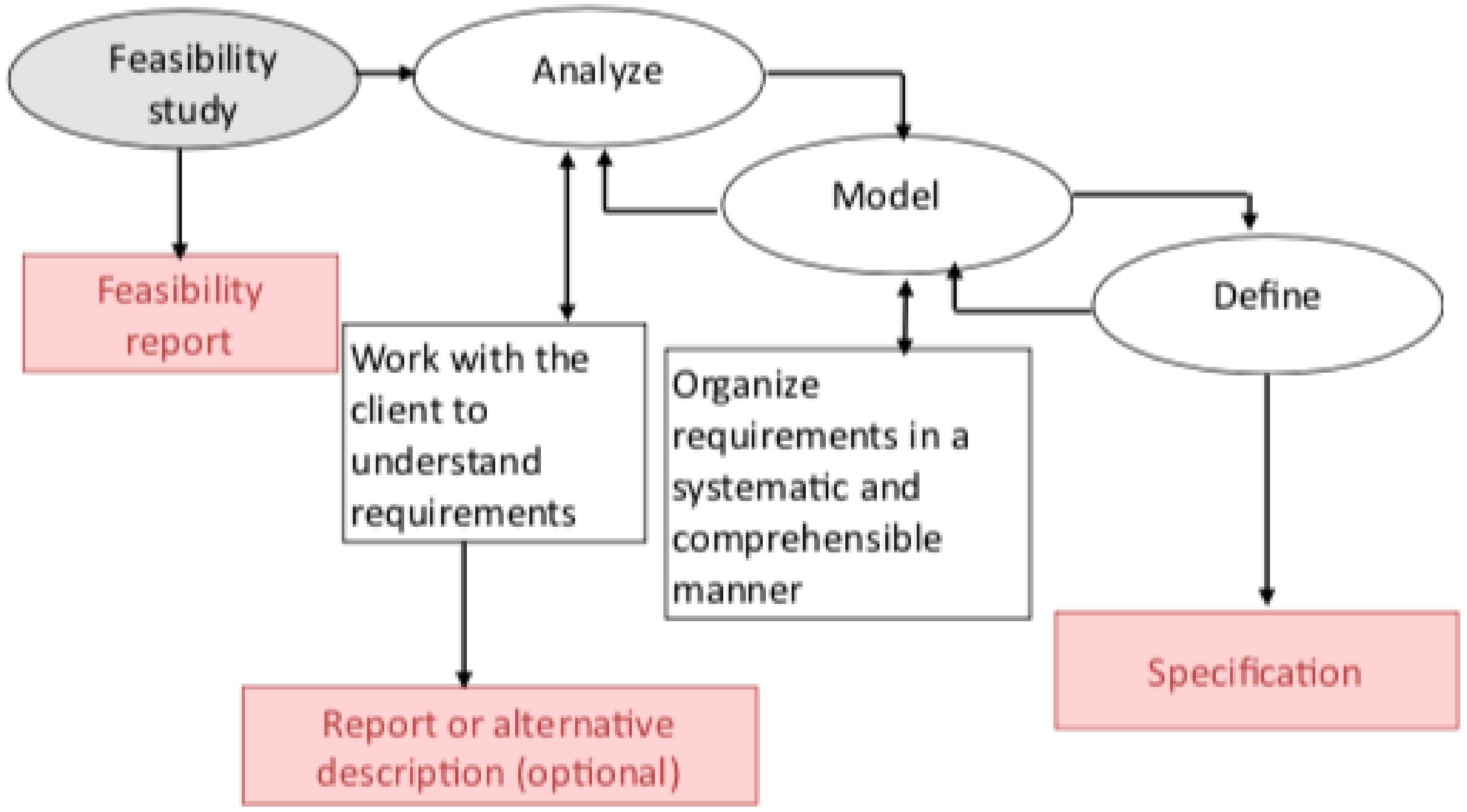
- **Modeling** to organize the requirements in a systematic and comprehensible manner

- **Define, record, and communicate** the requirements.

Heavyweight processes go through these steps for the entire system before beginning the design

With lightweight processes, these steps are done separately for each sprint.

Requirement Steps


Feasibility study
Analyze
Model
Define
Feasibility report
Work with the client to understand requirements
Organize requirements in a systematic and comprehensible manner
Report or alternative description (optional)
Specification

Requirement Dilemma

You cannot build a system unless you know what it is required to do

BUT…

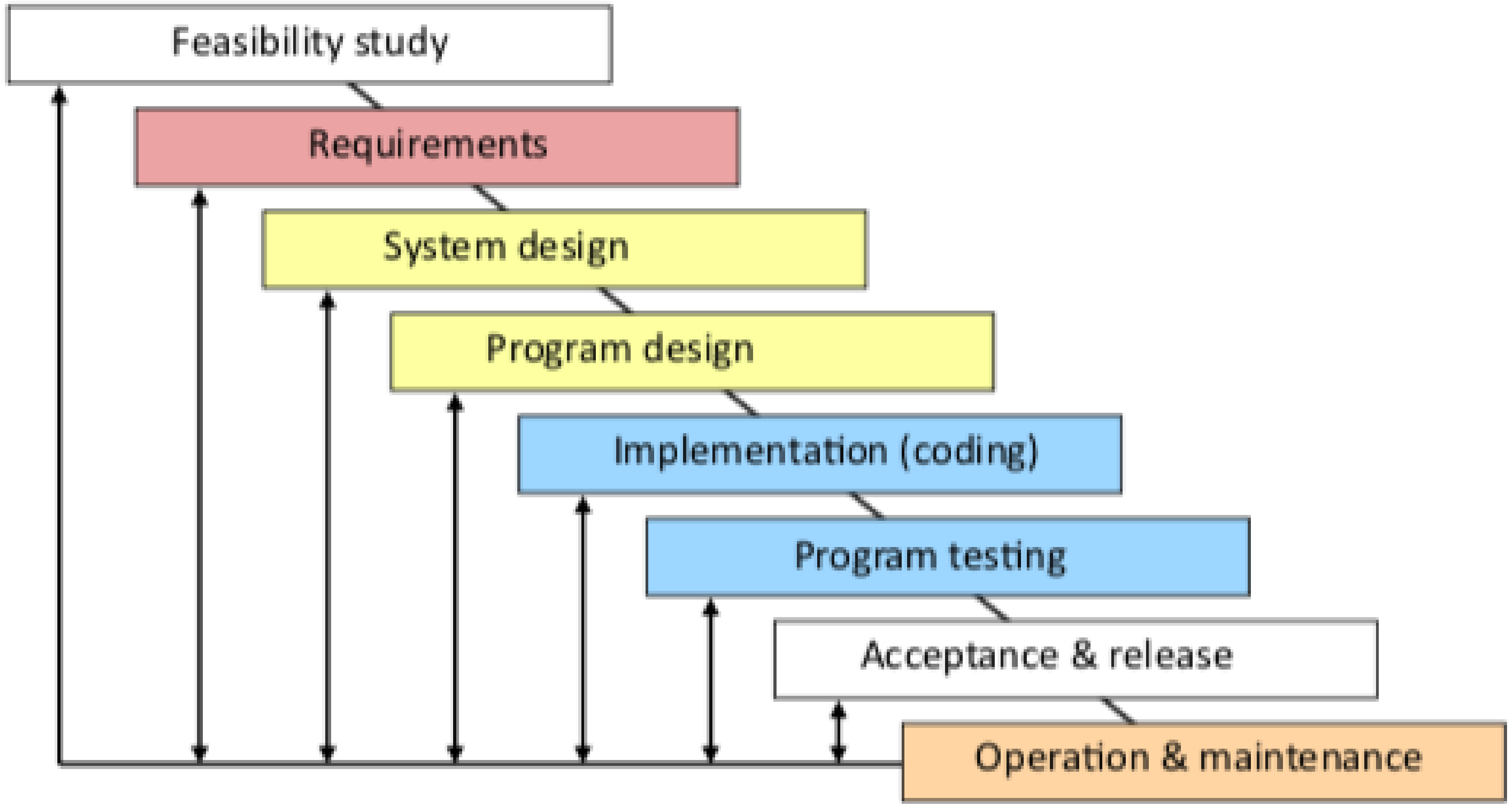
Clients may have only a partial understanding of requirements

Challenges

For clients:

- When they see the system, they ask for new features
- Frequently, they ask for major changes
- These changes may force you to rework large parts of the system
- These are problems for both **heavyweight** and **lightweight** processes.

Heavyweight Processes: Modified Waterfall Model



Requirements in Heavyweight Processes

Heavyweight processes expect detailed specification:

- Written document that specifies each requirement **in detail**
- Carefully checked by client and developers
- May be a contractual document
- Will be used for **acceptance testing**

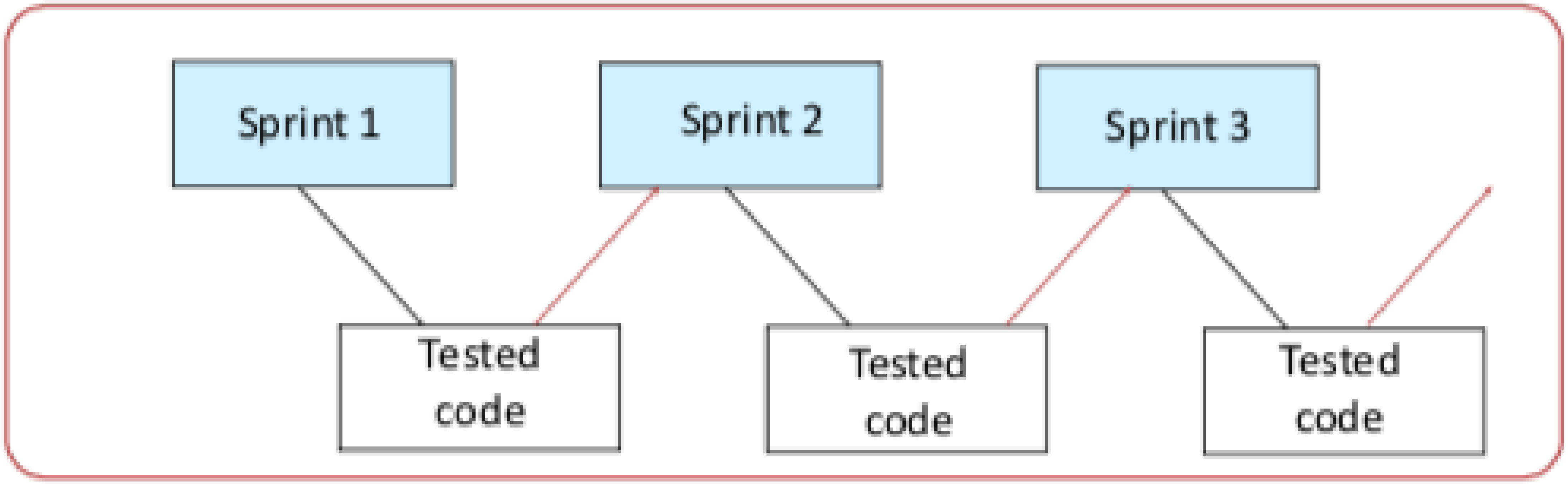
Difficulties:

- Specification is time consuming and difficult to create
- Specification is hard to maintain
- Checking a detailed specification is tedious
- Clients rarely understand the implications

The difficulty of ***creating*** *and* ***maintaining*** *a detailed requirements specification is one of the* ***reasons*** *that many organizations prefer* ***lightweight development processes***

Lightweight Processes: Agile Development

Each sprint has its own set of requirements



Requirements in Lightweight Processes (1)

Lightweight processes develop the requirements one sprint at a time:

- Working code is used for checking the requirements
- Client and developers work jointly on the requirements
- Minimal documentation is created during the sprint
- Fuller documentation is needed for future maintainers, but details are provided in the code

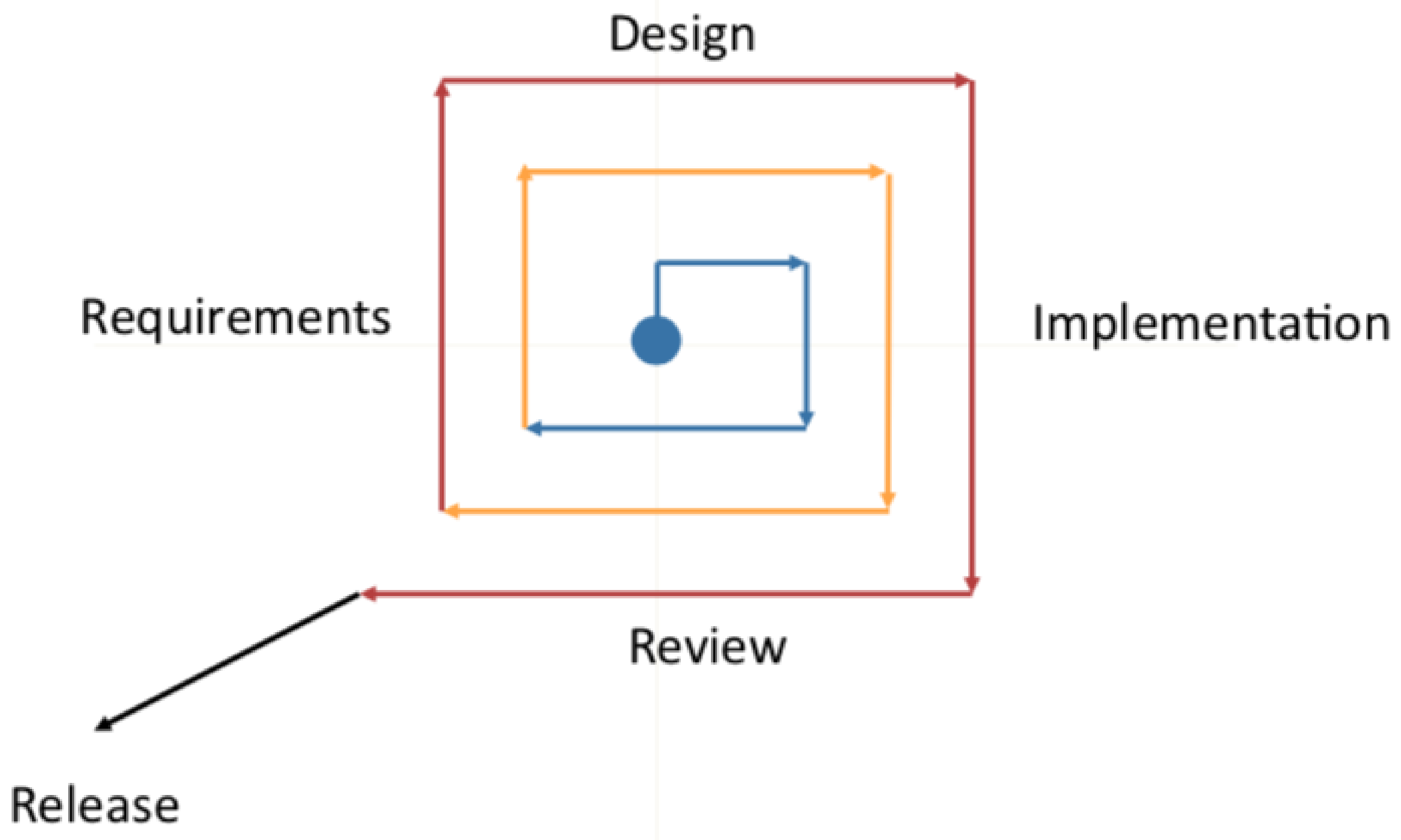
Requirements in Lightweight Processes (2)

Difficulties:

- Some requirements are system-wide and cannot be defined within a single sprint
  - e.g., data bases, security architectures, overall user interface design
- The requirements of future sprints may lead to major rework of earlier sprint

Middleweight Processes: Iterative Refinement

The requirements are revised for each iteration



Requirements in Middleweight Processes

Middleweight processes develop the requirements iteratively:

- The first iteration has an outline of the main requirements
- Each iteration refines the outline and add details
- Documentation is needed for future maintainers, but details are provided in the code

Difficulties:

- Each iteration may require major rework of previous work
- Developers often patch new requirements onto previous iterations

Discussion

- For a large system, you have to be flexible
- Both heavyweight processes and lightweight process have problems

BUT...

Both types of process work well, if used sensibly

- When using a **heavyweight** process, such as the modified waterfall model, specify the requirements in **moderate detail**, but be prepared for **revisions**. Some details can be left until later in the process

- When using a **lightweight** process, such as agile, develop **system-wide** requirements and the overall system architecture **early** in the process, perhaps before beginning the regular sprints

Requirement Goals

- **Understand** the requirements **in appropriate detail**

- **Ensure** that the **client and developers understand** the requirements and their **implications**

- **Define** the requirements in a manner that is **clear to the client**
  - This may be a written specification, prototype system, or other form of communication

- **Define** the requirements in a manner that is **clear to the people** who will **design, implement, and maintain the system**

Requirement Analysis: Interviews with Clients

Client interviews are the heart of the requirements analysis

Clients may have only a vague concept of requirements

- Allow plenty of time
- Prepare before you meet with the client
- Keep full notes
- If you do not understand, discuss and detail with client, again and again
- Repeat what you hear

Requirement Analysis: Understand the Requirements

Understand the requirements in depth

- Domain understanding
- Understanding the terminology
  - Clients often use specialized terminology. If you do not understand it, ask for an explanation
- Understanding of the real requirements of all stakeholders
  - Clients may not have clear ideas about what they require, or they may not express requirements clearly

Requirement Analysis: New and Old Systems

Clients often have an old system that is so familiar that they do not realize that it has functions that are not needed in a new system:

- A **replacement system** is when a system is built to replace an existing system
- A **legacy system** is an existing system that is not being replaced, but is being extended or must interface to a new system

In requirements analysis it is important to distinguish:

- features of the old system that are needed in the new system
- features of the old system that are not needed in the new system
- proposed new features

Requirement Analysis: Unspoken Requirements

Discovering the unspoken requirements is often the most difficult part of developing the requirements

Examples:

- Departmental friction, e.g., transfer of staff

Stakeholder Analysis

Identify the stakeholders

Who is affected by this system?

- Client
- Customers
- Users
- Senior management
- Administrators
- Computing staff

Example:

Web shopping site (shoppers, administration, finance, warehouse)

Viewpoint Analysis

Viewpoint Analysis

- Analyze the requirements as seen by each group of stakeholders
- Example: University Admissions System
  - Applicants
  - University administration
    - Admission office
    - Financial aid office
    - Special offices
  - Academic departments
  - Computing staff
  - Operations and maintenance

Specifying Requirements: Realism and Verifiability

Requirements must be **realistic**, i.e., it must be possible to meet them

- **Wrong**: The system must be capable of x (if no known computer system can do x at a reasonable cost)

Requirements must be **verifiable**, i.e., since the requirements are the basis for acceptance testing, it must be possible to test whether a requirement has been met

- **Wrong**: the system must be easy to use
- **Right**: After one day's training, an operator should be able to process 50 transactions per hour

Specifying Requirements: Communication

- With **heavyweight** processes, the requirements are defined by a full specification.
- With **lightweight** processes, the specification covers selected parts where there might be uncertainty

Objectives:

- Provide a basis for **acceptance testing**
- Provide **visibility**
- Be a foundation for system and program design
- Communicate with other teams who may work on or rely on this system or subsystem
- Inform future **maintainers**

Lightweight Processes (1)

With lightweight processes, experience and judgment are needed to distinguish between:

- **details** that can be left for later in the development process
- **key requirements** that must be agreed with the client early in the process

- A common fault is to miss crucial details
- This results in misunderstandings between client and the developers

The whole intent of lightweight processes is to have minimal intermediate documentation, but you need some

Lightweight Processes (2)

Lightweight processes use a outline specification + other tools

- Documentation describing key requirements in appropriate detail.
- Reviewed by client and developers.

Details provided by supplementary tools, e.g.,

- User interface mock-up or demonstration.
- Models, e.g., data base schema, state machine, etc.

Clients understand prototypes and models better than specification

- Iterative or incremental (agile) development processes allows the client to appreciate what the final system will do.

Functional requirements

Functional requirements describe the functions that the system must perform.

They include topics such as:

- Transactions
- Data
- User interfaces

Non-Functional requirements

Requirements that are not directly related to the functions that the system must perform

Product requirements

- performance, reliability, portability, etc…

Organizational requirements

- delivery, training, standards, etc…

External requirements

- legal, interoperability, etc…

Marketing and public relations

Non Functional Requirements - Examples

Example: Library Management System

Use technology that the client's staff are familiar with:

- Hardware and software systems (IBM/Unix)
- Database systems (Oracle)
- Programming languages (C and C++)

Requirement Analysis: Negotiation with Clients

Sometimes the client will request **functionality that is very expensive** or **impossible to implement**. Or two requirements may **contradict** each other.

This requires negotiation:

- Talk through the requirement with the client. Why is it wanted? Is there an alternative that is equivalent?
- Explain the reasoning behind your concern.
- Explain the technical, organizational, and cost implications.
- Be open to suggestions. Is there a gap in your understanding? Perhaps a second opinion might suggest other approaches.

The client and development team must resolve these questions.

Requirements vs. System Design

Technical decisions

- Requirements analysis should make **minimal assumptions** about the system design
- But the requirements definition must be **consistent** with **computing technology** and the **resources** available

In practice, analysis and design are interwoven. However:

- Do not allow assumptions about the design to influence the requirement analysis

7 – Requirement Analysis

(end of lecture)

ONE LOVE. ONE FUTURE.