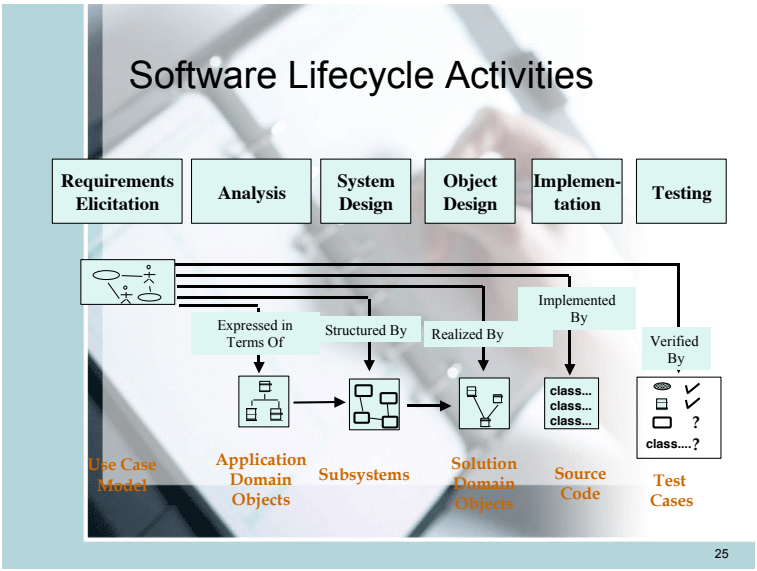
Introduction to Requirement Analysis

1

Today

- Overview of software lifecycle activities
- Understand and capture:
  - Functional requirements
  - Non-functional requirements
- How to write simple functional and non-functional requirement statements.
- Q & A

2



3

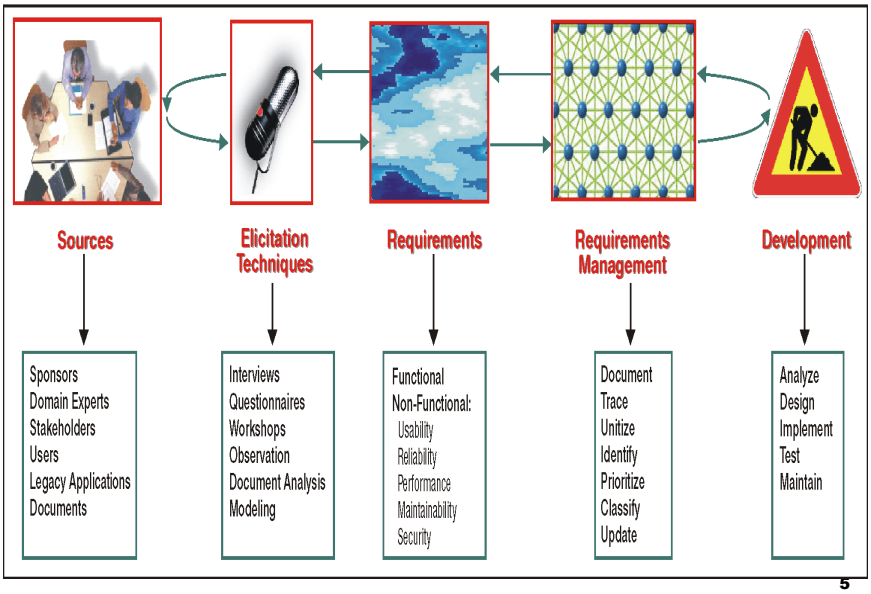
Requirements Gathering

- Collect and define all features in order to fulfill business objectives
- The reliability and the correctness of requirements is dependent on:
  - their sources,
  - techniques we use to elicit and verify
  - how effective we manage them

4

Requirements Gathering

An Ongoing Activity



5

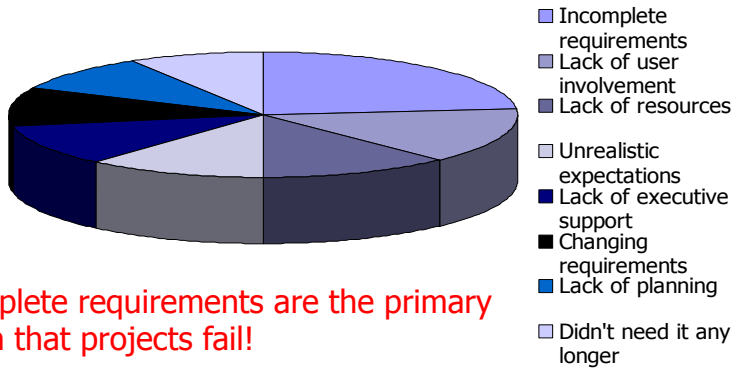
Purpose of Requirements

- To create a high-level specification of what should be implemented.
- To find out what the stakeholders (users, managers, installers, support personnel) need the system to do for them.

6

The importance of requirements

Project Failures



Incomplete requirements are the primary reason that projects fail!

The Standish Group

7

What are requirements?

- "A specification of what should be implemented."
- There are two types of requirements:
  - Functional requirements – what behavior the system should offer. Describes aspects of the system that are directly related to the functional behavior of the system.
  - Non-functional requirements - a specific property or constraint of the system. Describes aspects of the system that are not directly related to the functional behavior of the system.

8

Functional Requirement

- Describes the interactions between the system and its environment.
- What the system should do.
- Where do requirements come from?

9

Examples: Functional Requirement

- The ATM system shall check the validity of the inserted ATM card.
- The ATM system shall validate the PIN number entered by the customer.
- The ATM system shall dispense no more than $300 against any ATM card in any 24-hour period.

10

Non-Functional Requirement

- Describes aspects of the system that are not directly related to the functional behavior of the system.
- Non-functional constraints placed on the system.
- Four categories:
  - Usability
  - Reliability
  - Performance
  - Supportability

11

Examples: Non-Functional Requirement

- The ATM system shall be written in C++.
- The ATM system shall validate an ATM card in three seconds or less.
- The ATM system shall validate a PIN in two seconds or less.

12

Non-functional requirements: Usability

- **Usability:**
  - User learns to operate, prepare inputs and interpret outputs.
  - Defines how the behavior of the system meets the users and their work environment.
- Examples:
  - Conventions used for user interface
  - Scope of online help
  - Level of user documentation

13

Non-functional requirements: Reliability

- **Reliability:**
  - The ability of the system to perform its required functions under stated conditions.
  - Defines the dependency of the system regarding time, security and requirements.
- Examples:
  - Acceptable mean time to failure
  - Ability to detect faults
  - Ability to survive security attacks

14

Non-functional requirements: Performance

- **Performance:**
  - Concerned with quantifiable attributes of a system.
  - Attributes: response time, throughput and availability
- Examples of quantifiable attributes:
  - Accuracy – The acceptable error rate for the system should be 1 in 10,000,000 transactions.

15

Non-functional requirements: Supportability

- **Supportability/Maintainability:** Concerned with the ease of changes to the system after deployment.
- Examples:
  - Adaptability
  - Maintainability
  - Internationalization

16

Requirement Validation

- Completeness: All features of interest are described by requirements.
- Consistency: No two requirements of the specification contradict each other.

17

Requirement Validation (continued)

- Unambiguous: A requirement cannot be interpreted in two mutually exclusive ways.
- Accuracy: Describes the features of the system of interest to the client and developers intended to build. Alignment between model and developed application.

18

Requirement Specifications

- Realistic: if the system can be implemented within constraints.
- Verifiable: if repeatable tests can be designed to demonstrate that the system fulfills the requirements specification.
- Traceable: if each system function can be traced back to its corresponding set of requirements.

19

Writing requirements

- There is no UML standard way of writing requirements!
- Complete Functional Requirements statement - what the system should do
  - "The Part Replacement System shall provide the technician to record part removal from a customer engine."
- Non-functional Requirements - a constraint on how the functional requirements are implemented
  - "The ATM system shall authenticate a customer in four seconds or less"

<id> The <system> **shall** <function>

unique identifier — name of system — keyword — function to be performed

e.g. "R32 The ATM system **shall** validate the PIN number."

20

Summary

- We have seen how to capture:
  - Functional requirements
  - Non-functional requirements
- We have learned how to write simple functional and non-functional requirement statements.

21

To Learn More…

UML 2 and The Unified Process
Second Edition –
Jim Arlow and Ila Newstadt

Object-Oriented Systems Analysis and Design – *Ashrafi and Ashrafi*

22

Action Items

- Review the Procurement Case study prior to attending the next lecture.
- Review posted IP1 and case study

23