CAPTURING REQUIREMENTS

Mehwish Mumtaz

Requirements Engineering

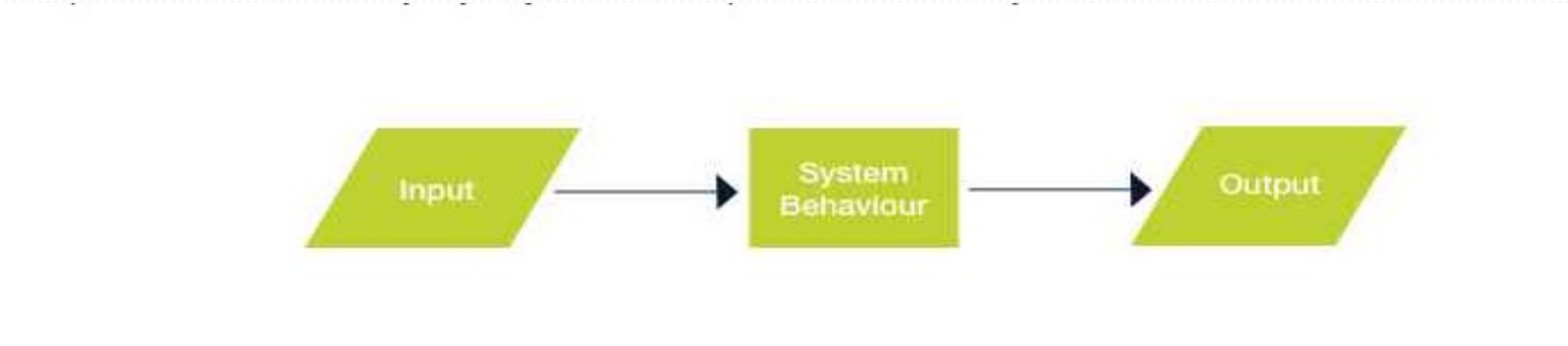
- Seven tasks:
  - Inception
  - Elicitation
  - Elaboration
  - Negotiation
  - Specification
  - Validation
  - Management

Types of Requirements

Functional Requirement

*"Any Requirement Which Specifies **What The System Should Do.**"*

In other words, a functional requirement will describe a particular behavior of function of the system when certain conditions are met, for example: "Send email when a new customer signs up" or "Open a new account".



Types of Requirements

Non-Functional Requirement

Non-functional requirements: These are basically the quality constraints that the system must satisfy according to the project contract. The priority or extent to which these factors are implemented varies from one project to other. They are also called non-behavioral requirements.

Functional VS Non-Functional Requirement

A functional requirement defines a system or its component.	A non-functional requirement defines the quality attribute of a software system.
It specifies “What should the software system do?”	It places constraints on “How should the software system fulfill the functional requirements?”
Functional requirement is specified by User.	Non-functional requirement is specified by technical peoples e.g. Architect, Technical leaders and software developers.
It is mandatory.	It is not mandatory.
It is captured in use case.	It is captured as a quality attribute.
Defined at a component level.	Applied to a system as a whole.
Helps you verify the functionality of the software.	Helps you to verify the performance of the software.
Functional Testing like System, Integration, End to End, API testing, etc are done.	Non-Functional Testing like Performance, Stress, Usability, Security testing, etc are done.
Usually easy to define.	Usually more difficult to define.
Example1) Authentication of user whenever he/she logs into the system. **2)** System shutdown in case of a cyber attack. **3)** A Verification email is sent to user whenever he/she registers for the first time on some software system.	**Example1)** Emails should be sent with a latency of no greater than 12 hours from such an activity. **2)** The processing of each request should be done within 10 seconds **3)** The site should load in 3 seconds when the number of simultaneous users are > 10000

Characteristics of Requirements

- **Unambiguous**
  - *REQ1 The system shall not accept passwords longer than 15 characters.*
  - The corrected requirement reflects the clarification:
  - The system shall display an error message if the user enters more than 15 characters.
  - *REQ1 The system shall not accept passwords longer than 15 characters. If the user enters more than 15 characters while choosing the password, an error message shall ask the user to correct it.*
- **Feasible**
  - The requirement should be doable within existing constraints such as time, money, and available resources:
  - *REQ1 The system shall have a natural language interface that will understand commands given in English language.*
  - This requirement may be not feasible within a short span of development time
- **Testable**
  - Testers should be able to verify whether the requirement is implemented correctly. The test should either pass or fail. To be testable, requirements should be clear, precise, and unambiguous. Some words can make a requirement untestable
  - Some adjectives: robust, safe, accurate, effective, efficient, expandable, flexible, maintainable, reliable, user-friendly, adequate
  - Some adverbs and adverbial phrases: quickly, safely, in a timely manner
  - Nonspecific words or acronyms: etc., and/or, TBD
  - *REQ1 The search facility should allow the user to find a reservation based on Last Name, Date, etc.*

Characteristics of Requirements

- **Consistent**
- There should not be any conflicts between the requirements. Conflicts may be direct or indirect. Direct conflicts occur when, in the same situation, different behavior is expected:
- *REQ1 Dates shall be displayed in the mm/dd/yyyy format.*
- *REQ2 Dates shall be displayed in the dd/mm/yyyy format.*
- Sometimes it is possible to resolve the conflict by analyzing the conditions under which the requirement takes place. For example, if REQ1 was submitted by an American user and REQ2 by a French user, the preceding requirements may be rewritten as follows:
- *REQ1 For users in the U.S., dates shall be displayed in the mm/dd/yyyy format.*
- *REQ2 For users in France, dates shall be displayed in the dd/mm/yyyy format.*
- This can eventually lead to the following requirement:
- *REQ3 Dates shall be displayed based on the format defined in the user's web browser.*

Characteristics of Requirements

- **Correct**
- The requirement meets the actual business or system need. An incorrect requirement can still be implemented resulting in a business process or system that does not meet the business needs.
- **Complete**
- Each requirement must fully describe the functionality to be delivered. The individual requirement is not missing necessary or relevant information.
- **Relevant**
- **Traceable**
- *REQ1 The system shall provide the opportunity to book the flight, purchase a ticket, reserve a hotel room, reserve a car, and provide information about attractions.*
- has a unique ID, it is usually traceable.

Testable Requirements

- Testable/Measurable Requirement:
- Testable requirements are helpful in making good design
- Requirements that are not testable are likely to be ambiguous, incomplete and incorrect

Testable Requirements

- Robertson and Robertson suggest 3 ways to help make requirements testable:
  - Specify a quantitative description for each adverb and adjective
  - Replace pronouns with specific names of entities
  - Make sure that every noun is defined in exactly one place in the requirements document

Testable Requirements

- Some examples:
  - Not Testable: Water quality information must be accessible immediately
  - Testable: Water quality information must be retrieved within five seconds of request
  - Not Testable: The system should handle a large number of users at a time
  - Testable: The system should handle 5000 users at a time
  - Not Testable: User should press the Save button when writing text in the system. This prevents it from being lost.
  - Testable: User should press the Save button when writing a note in the system. Pressing the Save button prevents the text from being lost.

Requirements Documentation

Requirement Definition: Steps Documenting Process

- Outline the general purpose and scope of the system, including relevant benefits, objectives, and goals
- Describe the background and the rationale behind proposal for new system
- Describe the essential characteristics of an acceptable solution
- Describe the environment in which the system will operate
- Outline a description of the proposal, if the customer has a proposal for solving the problem
- List any assumptions we make about how the environment behaves

Requirements Documentation

IEEE Standard for SRS Organized by Requirements

1. Introduction to the Document
   - 1.1 Purpose of the Product
   - 1.2 Scope of the Product
   - 1.3 Acronyms, Abbreviations, Definitions
   - 1.4 References
   - 1.5 Outline of the rest of the SRS
2. General Description of Product
   - 2.1 Context of Product
   - 2.2 Product Functions
   - 2.3 User Characteristics
   - 2.4 Constraints
   - 2.5 Assumptions and Dependencies
3. Specific Requirements
   - 3.1 External Interface Requirements
     - 3.1.1 User Interfaces
     - 3.1.2 Hardware Interfaces
     - 3.1.3 Software Interfaces
     - 3.1.4 Communications Interfaces
   - 3.2 Functional Requirements
     - 3.2.1 Requirement 1
     - 3.2.2 Requirement 2
     - ...
   - 3.3 Performance Requirements
   - 3.4 Design Constraints
   - 3.5 Other Quality Requirements
   - 3.6 Other Requirements
4. Appendices

Functional Requirements for a Spellchecker:

Requirement 1:
System should keep track of words being typed in the existing word processor

Requirement 2:
System should check against each typed word if it is a valid word in the corresponding dictionary

Requirement 3:
System should highlight the wrong words (i.e. which are not part of dictionary) while the user types in the existing word processor.

Requirement 4:
System should give the user an option to make correction in case there is a wrong word typed

Requirement 5:
System should suggest correct words when user opts to make a correction against a wrong word

Non Functional Requirements for a Spellchecker:

Requirement 1:
System should integrate with an existing word processor that runs on Linux.

Requirement 2:
System should use Red color to highlight the wrong words

Requirements Documentation

Process Management and Requirements Traceability

- Process management is a set of procedures that track
  - the requirements that define what the system should do
  - the design modules that are generated from the requirement
  - the program code that implements the design
  - the tests that verify the functionality of the system
  - the documents that describe the system
- It provides the threads that tie the system parts together
- Numbering the requirements.

Validation and Verification

Remember that the requirements documents serve both as a contract between us and the customer, detailing what we are to deliver, and as guidelines for the designers, detailing what they are to build. Thus, before the requirements can be turned over to the designers, we and our customers must be absolutely sure that each knows the other's intent, and that our intents are captured in the requirements documents. To establish this certainty, we *validate* the requirements and *verify* the specification.

Validation and Verification

- In requirements validation, we check that our requirements definition accurately reflects the customer's needs
- In verification, we check that one document or artefact conforms to another e.g code conforms to design .design conforms to SRS.
- Verification ensures that we build the system right, whereas validation ensures that we build the right system

Validation and Verification

Validation	Walkthroughs
	Reading
	Interviews
	Reviews
	Checklists
	Models to check functions and relationships
	Scenarios
	Prototypes
	Simulation
	Formal inspections
Verification	Cross-referencing
	Simulation
	Consistency checks
	Completeness checks

Validation and Verification

Requirements Review for requirement validation

- Client representative and our staff examine the SRS document to identify the problems e.g. design team ,testers
- Review the stated goals and objectives of the system
- Compare the requirements with the goals and objectives
- Review the environment in which the system is to operate
- Review the information flow and proposed functions
- Assess and document the risk, discuss and compare alternatives
- Testing the system: how the requirements will be revalidated as the requirements grow and change
- E.g who will provide the test data
- Which requirements in which phases

Validation and Verification

Verification

- Check that the requirements-specification document corresponds to the requirements-definition
- Make sure that if we implement a system that meets the specification, then the system will satisfy the customer's requirements
- Ensure that each requirement in the definition document is traceable to the specification

Measuring Requirements

- Measurements focus on three areas
  - product
  - process
  - resources
- Number of requirements can give us a sense of the size of the developed system
- Number of changes to requirements
  - Many changes indicate some instability or uncertainty in our understanding of the system

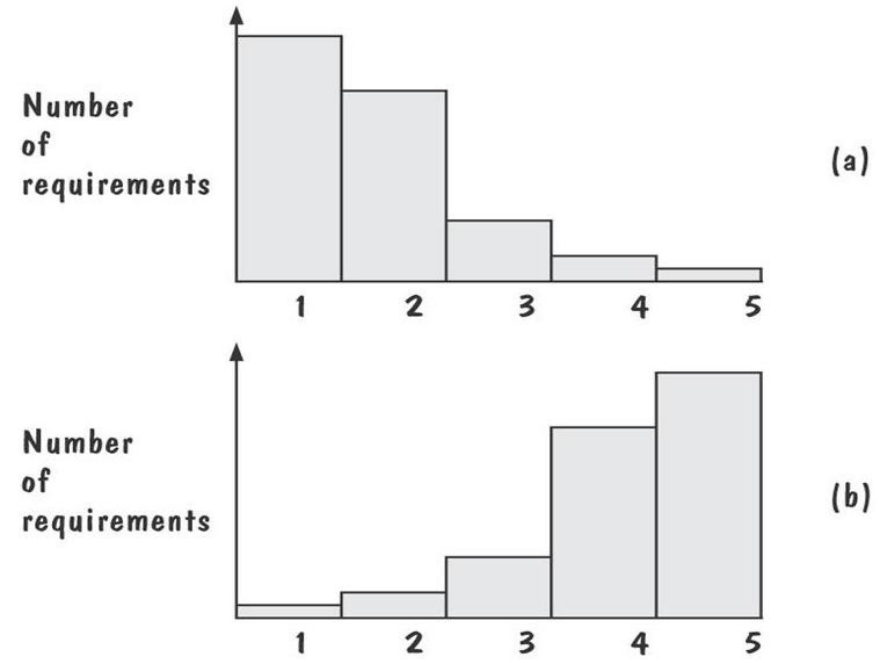
Measuring Requirements

Rating Scheme on Scale from 1 to 5

1. You understand this requirement completely, have designed systems from similar requirements, and have no trouble developing a design from this requirement
2. Some elements of this requirement are new, but they are not radically different from requirements that have been successfully designed in the past
3. Some elements of this requirement are very different from requirements in the past, but you understand the requirement and can develop a good design from it
4. You cannot understand some parts of this requirement, and are not sure that you can develop a good design
5. You do not understand this requirement at all, and can not develop a design

Measuring Requirements: Testers/Designers Profiles

- Figure (a) shows profiles with mostly 1s and 2s
  - The requirements are in good shape
- Figure (b) shows profiles with mostly 4s and 5s
  - The requirements should be revised



Elaboration

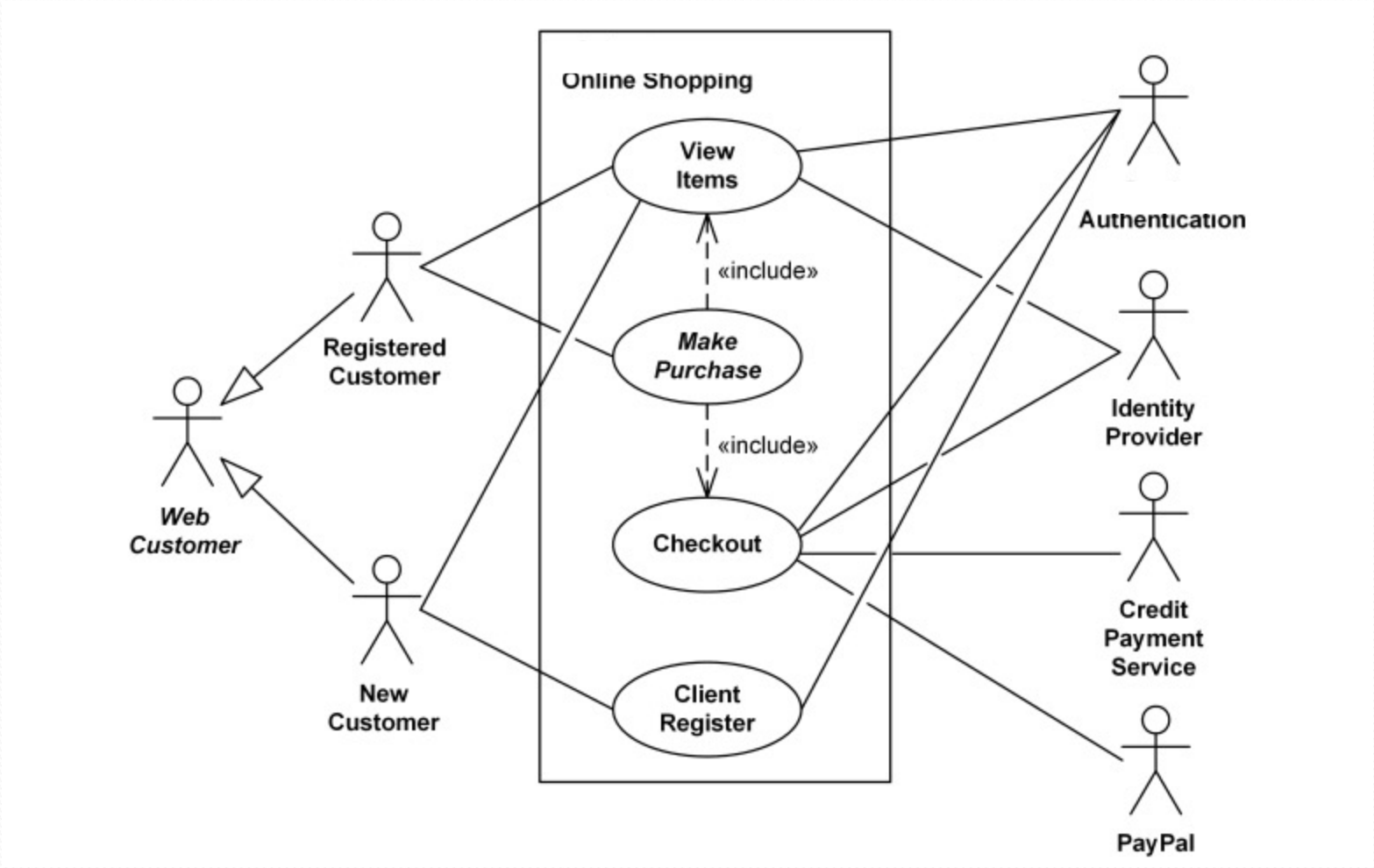
- Analyze, model, specify…
- Some Analysis Techniques
  - **Data Flow Diagrams (DFD)**
  - **Use case Diagram**
  - Object Models (ER Diagrams, Abstract class diagrams)
  - Decision Tables
  - State Diagrams (Statecharts)
  - Fence Diagrams
  - Event Tables
  - Petri Nets
  - Event Traces (Message Sequence Charts)

Flow-oriented, Scenario-based, Class-based, Behavioral

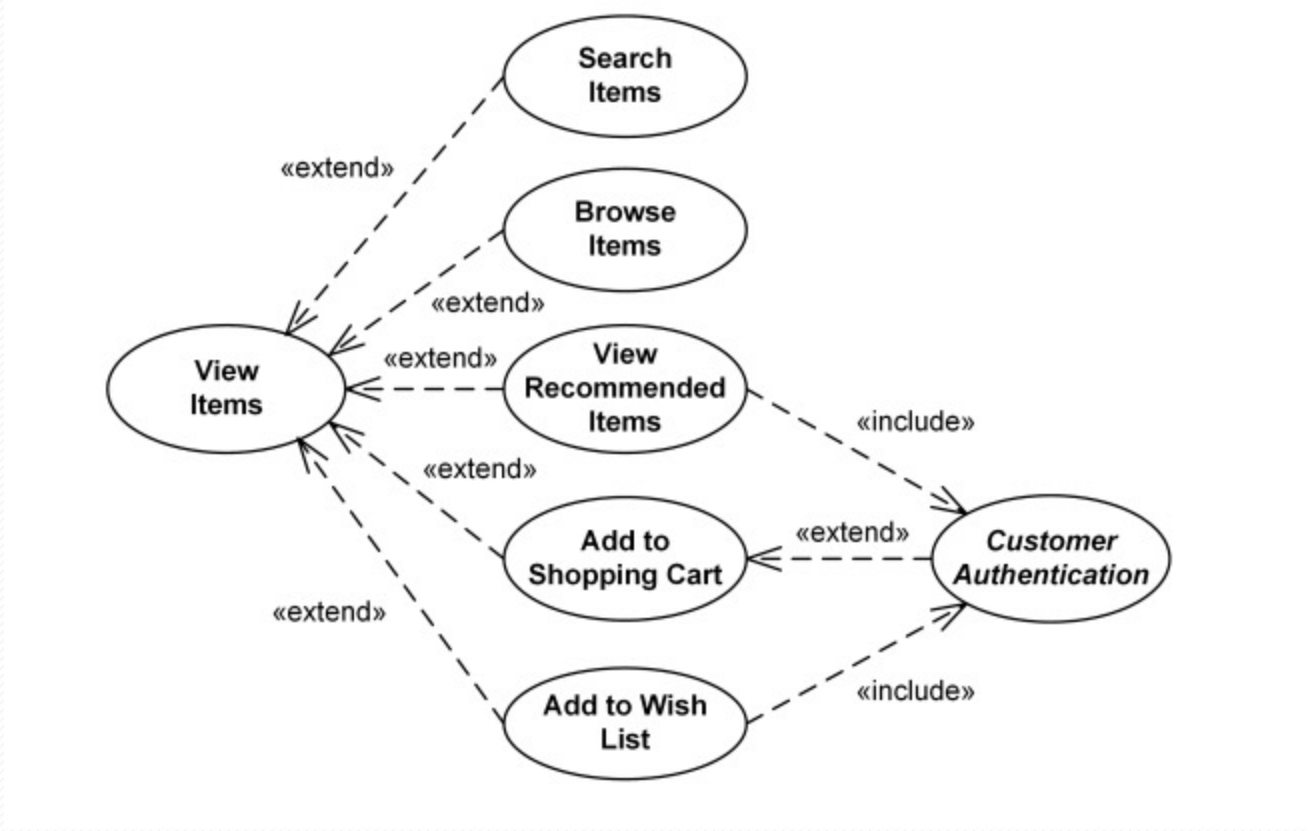
Use Case Diagram

- Components
  - A large box: *system boundary*
  - Stick figures outside the box: *actors*, both human and systems
  - Each oval inside the box: a use case that represents some major required functionality and its variant
  - A line between an actor and use case: the actor participates in the use case
- Use cases do not necessarily model all the tasks, instead they are used to specify user views of essential system behaviour.

Use Case Diagram



Use Case Diagram



Use Case Diagram