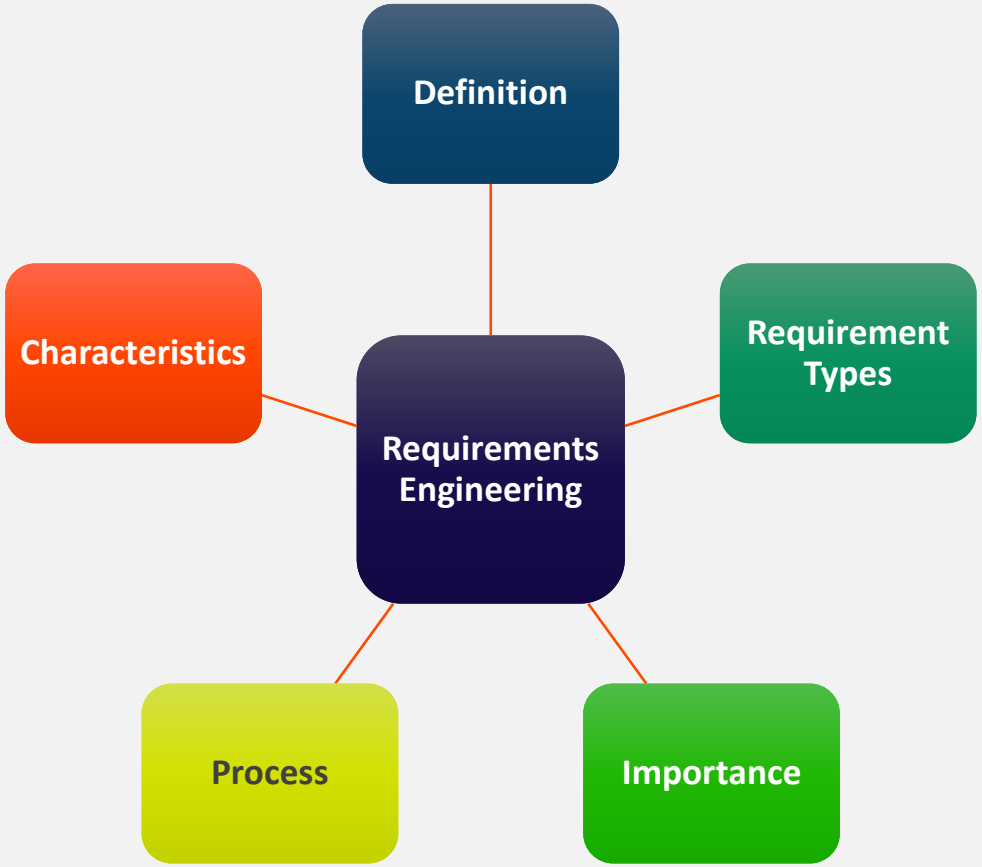
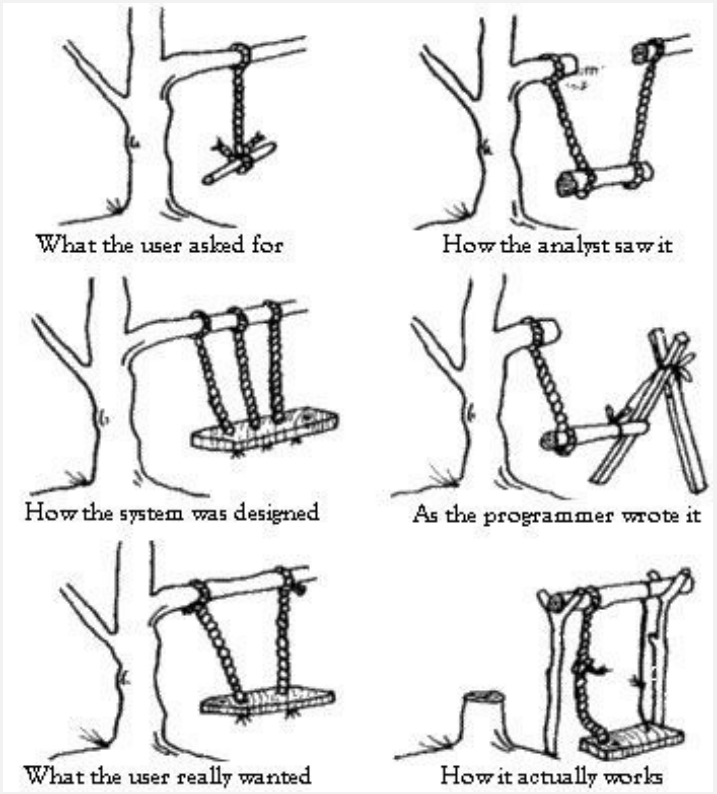
# CSEN603 – Software Engineering

## Lecture 2: Requirements Engineering

Mervat Abuelkheir
Ammar Yasser
Mohamed Agamia
Nada Hisham




# Software Engineering

- Architecture
- Requirements Engineering
- Design and Design Patterns
- Implementation
- Verification and Validation
- Quality and Maintenance
- Scale and Evolution
- Economics

Process, Models, Methods

# What are Requirements?

- **Requirements** define the **functions** of the system from the client's viewpoint and the **constraints** under which the system operates and is developed
- Requirements are essential for **software development**
- Requirements form the basis for **acceptance testing**
  - **Essential contributor to SW product quality**

# Requirements Engineering

- **Requirements engineering** involves discovering, analyzing, documenting, and maintaining requirements and constraints
- **Constraints** are generated during the requirements engineering process
- Requirements may be developed in a self-contained study, or emerge incrementally
- The software development team and the client need to WORK TOGETHER CLOSELY DURING THE REQUIREMENTS PHASE

## Clients/Stakeholders

Any **person** or **organization** who is affected by the system in some way and so has a legitimate interest in the system

**Stakeholder types**

- End users
- System managers
- System owners
- External stakeholders

Requirements must be developed in a manner that is **understandable** by both the client and the development staff

# Requirements

**SCOPE OF WORK IN BRIEF (DETAIL IN ANNEXURE)**

**Broad scope of the project is as follows:**

1. Defining and documenting the architecture and the detail design / development for the new web platform for propagation, capturing details of students, movement of student from one stage to another stage, training, membership and thereafter services like Placement etc. Analysis of Existing vs. Proposed system in respect of value addition / advantage. Design should be structured such that most of changes like enabling / disabling of any service should be through User Interface.

2. Defining the hardware specification for the new web platform. Creating detailed project deliverable documents (User Scenarios and workflows, User Requirements Specification, Detail Design Document, Test Case documents etc.).

3. The Software application / CMS / Online Portal should be based on MS.Net and MSSQL as backend with no additional license cost and unlimited logins.

4. The New Web Application should follow the SOA architecture, so that functionalities of every module should be Service dependent. The application should have layer based architecture that is flexible enough to have limited impact changes throughout the layers of the application. The architecture must demonstrate loose coupling across layers, and must list flexibility constraints, if any.

5. Modules for stakeholders like Student Services, Examination, Training, Licentiate, Membership, Placement, Sale of Material, Regional Offices/ Chapter office Program Management / Seminars & Trainings, Financial Management (only of fees received) / Reconciliation for individual offices of fees received from stakeholders etc. should be integrated in a single web application. Privileges to pay fees with / without login as per the controls with Administrator.

6. All the history / existing data of the Institute should migrate in the new system. There should not be any history / existing data loss.

7. Support & Migration of data during a parallel run: At the time of Production Deployment both applications (Existing and New) will be run parallel for a certain time period for smooth Functionality Testing. After successful verification and approval the existing application will be sunset and the data entered during the pilot run will also have to be migrated to New System.

8. Provision for all entitlements / privileges for all types of Profiles like as End Users, Super Users, IT Users, IT Administrators, DB Administrator and Super Administrator with password assistance through SMS and email. Role based access and authorization of various modules.

9. Report Requirement
    a. The Application should contain a Dashboard of tailor made MIS pages which shows output in the form of maps, charts with threshold limits.
    b. There should be dynamic report generation / analysis feature within the web application. User should be able to select required columns, filtration, sorting at runtime for any module according to their access. The template can be saved for future use also.
    c. Authoring and Maintaining templates for Email / SMS for each trigger (action against any updation). The Institute has already hired a Bulk SMS and Bulk Mailing System / facility from third party vendor, which is to be integrated by the vendor for this purpose. Further this is informed that the vendor for Bulk SMS and Bulk Mailing systems of the Institute may change in future. In view of the same the vendor has to design and develop the system such that it may cater to the change of the system of new form with few configurations here and there which may be documented and provided to the Institute.

10. The Application should pass OWASP Top10 security check certificate to stop hacking attempts before going LIVE. A CERT-In empanelled auditor must certify that the application is found to be free of all OWASP Top 10 vulnerabilities. Managing security of the new system from all type of

33. The proposed application should use Social Media and user moderated Discussion Boards. The vendor is to propose how interaction with Social Media can benefit the application. Preferably Integration with Face book, Twitter etc.

34. The software application / processes have to run / interact with stakeholders without or minimal human intervention. To achieve the same all validation controls has to be imbibed in the software with provision to configure the rules.(e.g. City University, State, Country validation,(if Other then adding in the master after approval), cross validation of city/ pin code/ chapter of ICSI)( candidates will be shown the options they deserve)

35. The new Software Application should be able to communicate with the Institute's Sharepoint site for fetching / placing some files (images / documents) and processed afterwards. The firm maintaining a separate file server shall hold the responsibility of data migration to this repository.

36. The software application should be designed in such a manner so that it would cater below listed Strategic Action Points:
    a. Live chat tool before and after login.
    b. Integrating / Monitoring grievance / call center
    c. Surveys
    d. Feedback monitoring with reminders (SMS/Email) based on rules for e.g. standard tasks are expected on monthly / weekly basis.
    e. Discussion board
    f. Subject wise Video / PPT repository before / after login with role based access

37. The system should be open to be integrated with any third party application in future.

38. The system should be able to link with existing Files server where contents like Photos, Certificate copy is present.

39. The software application should have the provision to capture attendance of employees from Chapter Offices where Biometric Machine has not been installed, in such as way so that the captured attendance should be approved from respective Chairmen / Executive Officer from that Chapter.

https://www.icsi.edu/media/webmodules/tender/Development_and_Implementation_of_Web_Based_Application_Software_Tender.pdf

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"**

*Source: Standish Group*

# Requirements Engineering

## Importance

# Requirements Types

- **User Requirements**
  - Statements and 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

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.

# Requirements Types

- **Functional**
  - 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
- **Nonfunctional**
  - **Quality** characteristics and attributes
  - **Constraints** on the services or functions offered by the system such as timing constraints, constraints on the development process, standards, etc.

## Functional Requirements

- 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**
- **Examples**: transactions, data, UI

# Requirements Are Tricky – Requirements Imprecision

| Example Functional Requirement | User Intention | Developer Interpretation |
| --- | --- | --- |
| A user shall be able to search the appointments lists for all clinics | Search for a patient name across all appointments in all clinics | Search for a patient name in an individual clinic. User chooses clinic then search |

**Ambiguous requirements may be interpreted in different ways by developers and users – Problems!**

# Requirements Types

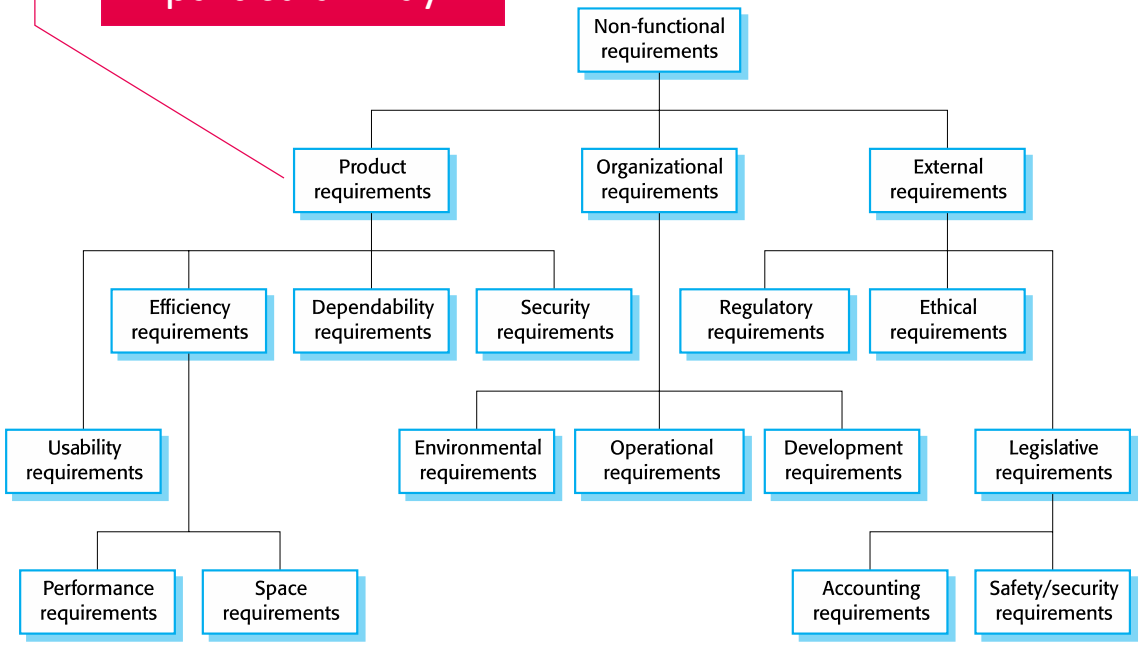
- Functional
  - Statements of **services the system should provide**, **how the system should react to particular inputs** and **how the system should behave in particular situations**
  - May state what the system should not do
- **Nonfunctional**
  - **Quality** characteristics and attributes such as reliability, response time, etc.
  - **Constraints** on the services or functions offered by the system such as timing constraints, constraints on the development process, standards, etc.

## Nonfunctional Requirements

- **Constraints** are I/O device capability, system representations, storage capacity, etc.
- **Process requirements** may be specified mandating a particular IDE, programming language or development method
- May be **more critical than functional requirements**. If not met, the system may be useless

# Requirements Types

specify that the delivered product must behave in a particular way

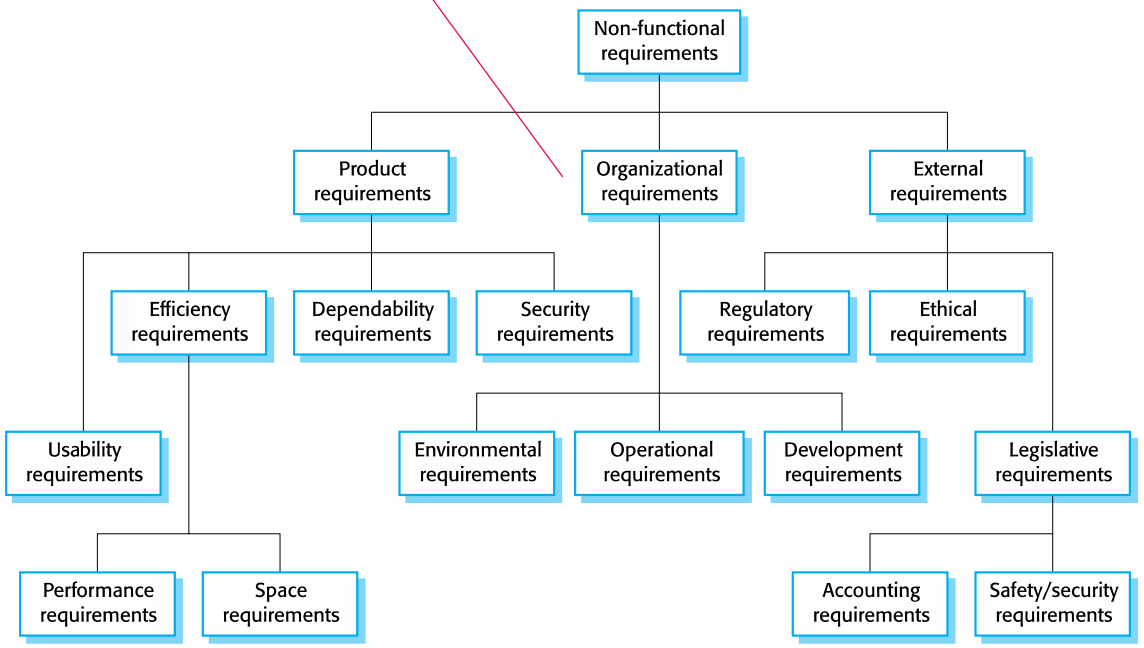



## Nonfunctional Requirements

- May affect the overall architecture of a system rather than the individual components
- **A single nonfunctional requirement**, such as a security requirement, **may generate a number of related functional requirements** that define the system services required

# Requirements Types

a consequence of organizational policies and procedures

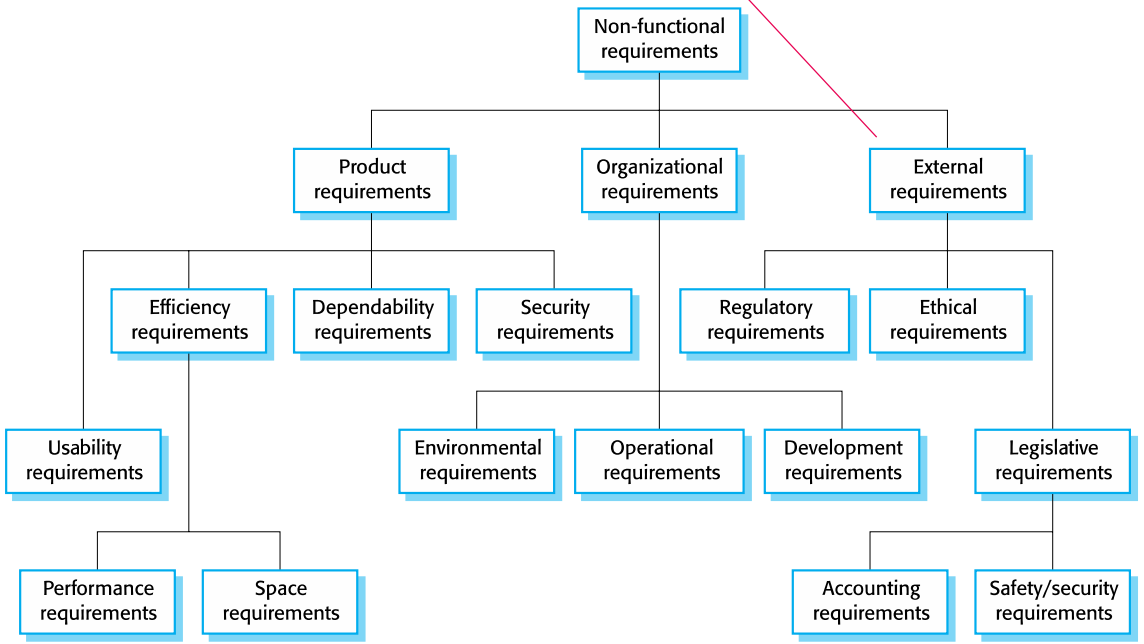



## Nonfunctional Requirements

- May affect the overall architecture of a system rather than the individual components
- **A single nonfunctional requirement**, such as a security requirement, **may generate a number of related functional requirements** that define system services that are required

# Requirements Types

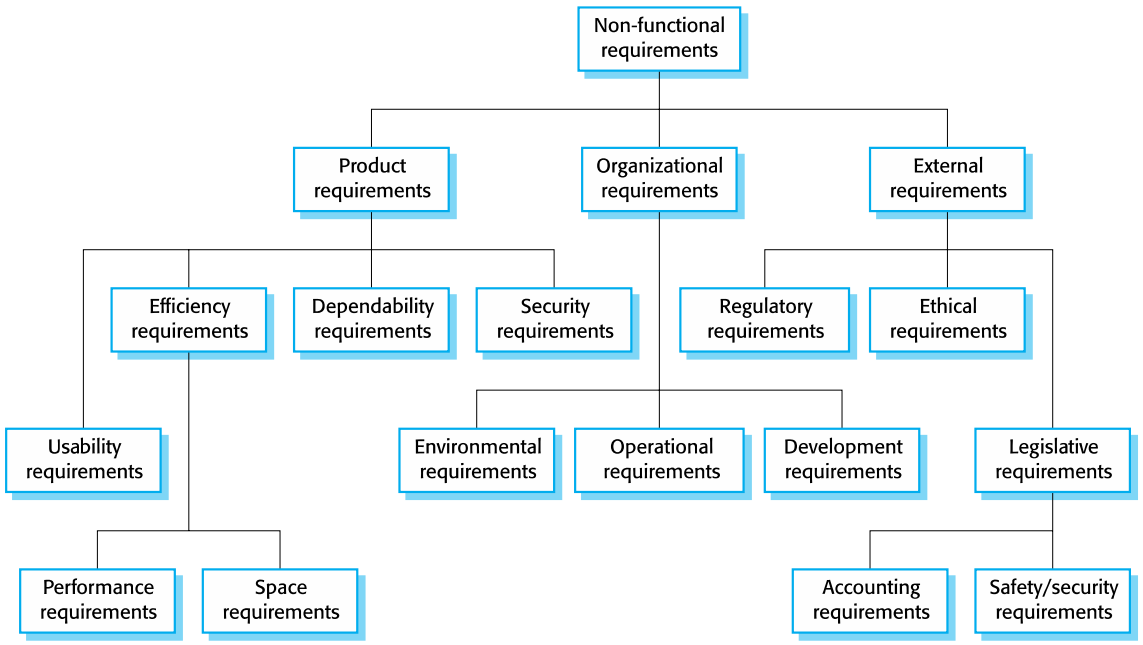
arise from factors which are external to the system and its development process




## Nonfunctional Requirements

- Nonfunctional requirements may be very **difficult to state precisely**
- Imprecise requirements may be **difficult to verify**
- **Metrics** may help!

# Requirements Types

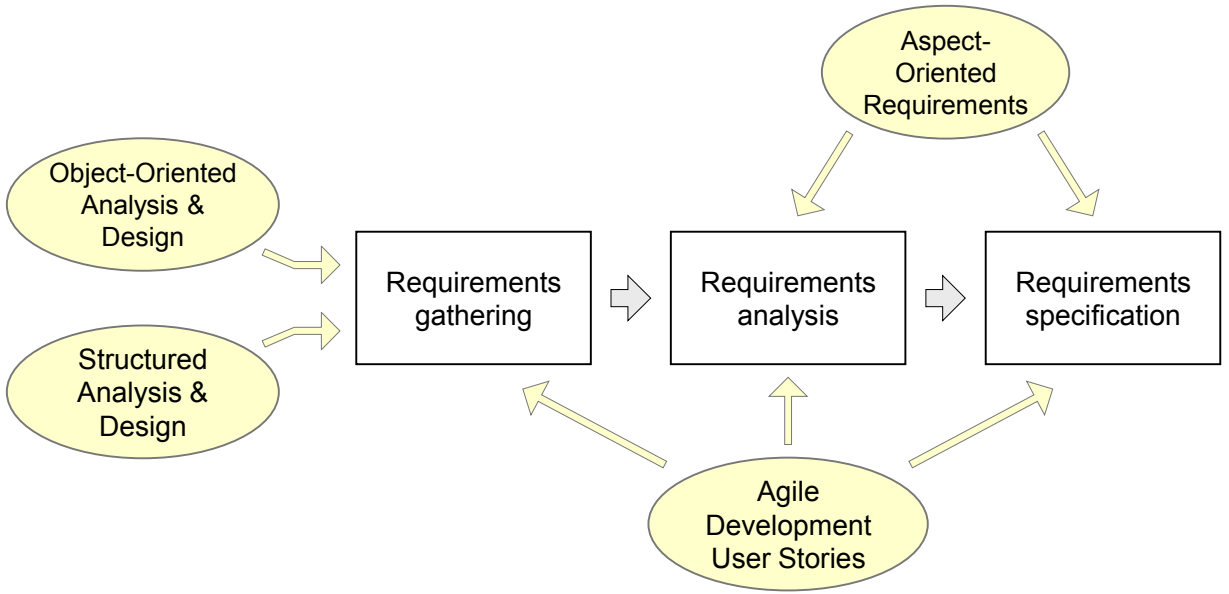



## Metrics for Quantifying Nonfunctional Requirements

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

# Requirements Engineering Process

- Requirements **development**
  1. **Gathering/Elicitation**
     - Statement of needs
     - Feasibility study
  2. **Analysis, Modeling, and Negotiation**
     - Structure – ERD and DFD
     - OO
       - Scenario – activity diagram, use case diagram
       - Behavior – state diagram, sequence diagram
       - Class – CRC, class diagram
  3. **Specification** of information, function, behavior
     - Build SRS
     - Official documents
  4. **Verification and validation**
     - Check product meets need
- Requirements **management**




- With classical software development methods (*waterfall*), you **do requirements engineering once**
- With software development methods (*spiral* or *agile*), you **repeat requirements engineering stages several times**

[more on waterfall, agile, and spiral later in course]

# Step 1 – Requirements Elicitation/Gathering

## Requirements Elicitation Methods

### Interviewing

- **Closed interviews**: predetermined list of questions
- **Open interviews**: explore various issues with stakeholders
- Be open-minded, avoid pre-conceived ideas about the requirements, **listen**
- Prompt interviewee to get discussions going using a **requirements proposal**, or by working together on a **prototype system**

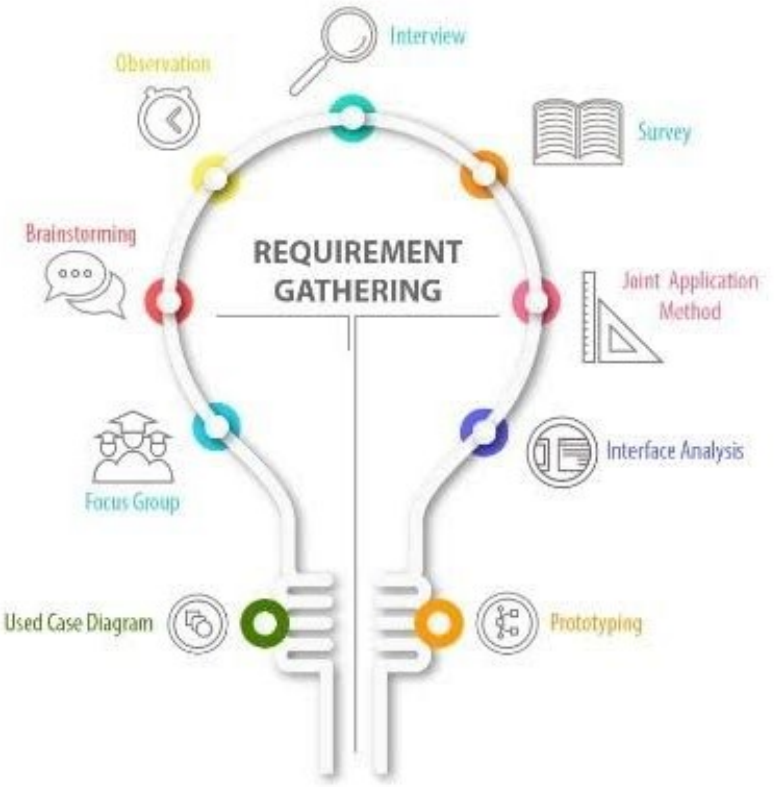
### Challenges

- Time-consuming
- Needs lots of preparation before interviews
- Requirements engineers may not understand specific domain terminology

➢ Prepare well and allocate enough time for each interview

➢ Small group meetings are more effective

➢ It helps to repeat what you hear



# Requirements Elicitation Methods (Cont.)

## User Stories

- Real-life examples of how a user interacts with a system
- Describes how a system may be used for a particular task
- Stakeholders can relate to stories and can comment on their situation with respect to the story
- Follows template:

"As a [persona], I [want to], [so that]."

## Scenarios

- A **structured form** of user story
- Scenarios should include:
  - Description of **starting situation**
  - Description of the normal **flow of events**
  - Description of **what can go wrong**
  - Information about other **concurrent activities**
  - Description of **state** when scenario **finishes**

# Developing a Scenario with a Client

## Example

Requirements are being developed for a system that will enable university students to take exams online from their own rooms using a web browser

## User Story

"As a **student**, I want to **be able to take my exams online**, so that **I can answer exam questions at the convenience of my own room**."

Who are we building this for? **Persona**

What is it they're actually trying to achieve? **Intent**

What's the overall benefit they're trying to achieve?
What is the big problem that needs solving? **Goal**

# Developing a Scenario with a Client

## Example

Requirements are being developed for a system that will enable university students to take exams online from their own rooms using a web browser

- **Purpose**: Scenario that describes the use of an online Exam system by a representative student
- **Individual**: [Who is a typical student?] Student $A$, senior at GUC, major in CS. [Do other universities differ?]
- **Equipment**: Any computer with a supported browser. [Is there a list of supported browsers? Are there any network restrictions?]

## Scenario

1. Student $A$ authenticates. [How does a GUC student authenticate?]
2. Student $A$ starts browser and types URL of Exam system. [How does the student know the URL?]
3. Exam system displays list of options. [Is the list tailored to the individual user?]
4. Student $A$ selects CSEN603 Exam 1
5. A list of questions is displayed, each marked to indicate whether completed or not. [Can the questions be answered in any order?]

# Developing a Scenario with a Client

## Example

Requirements are being developed for a system that will enable university students to take exams online from their own rooms using a web browser

- Purpose: Scenario that describes the use of an online Exam system by a representative student
- Individual: [Who is a typical student?] Student **A**, senior at GUC, major in CS. [Do other universities differ?]
- Equipment: Any computer with a supported browser. [Is there a list of supported browsers? Are there any network restrictions?]

## Scenario (cont.)

6. Student ***A*** selects a question and chooses whether to submit a new answer or edit a previous answer. [Is it always possible to edit a previous answer? Are there other options?]
7. [What types of question are there: text, multiple choice, etc.?] The first question requires a written answer. Student ***A*** is submitting a new answer. The student has a choice whether to type the solution into the browser or to attach a separate file. Student ***A*** decides to attach a file. [What types of file are accepted?]

# Developing a Scenario with a Client

## Example

Requirements are being developed for a system that will enable university students to take exams online from their own rooms using a web browser

- Purpose: Scenario that describes the use of an online Exam system by a representative student
- Individual: [Who is a typical student?] Student **A**, senior at GUC, major in CS. [Do other universities differ?]
- Equipment: Any computer with a supported browser. [Is there a list of supported browsers? Are there any network restrictions?]

## Scenario (cont.)

8. For the second question, the student chooses to edit a previous answer. Student ***A*** chooses to delete a solution previously typed into the browser, and to replace it with an attached file. [Can the student edit a previous answer, or must it always be replaced with a new answer?]
9. As an alternative to completing the entire exam in a single session, Student ***A*** decides to saves the completed questions to continue later. [Is this always permitted?]

# Developing a Scenario with a Client

## Example

Requirements are being developed for a system that will enable university students to take exams online from their own rooms using a web browser

- Purpose: Scenario that describes the use of an online Exam system by a representative student
- Individual: [Who is a typical student?] Student **A**, senior at GUC, major in CS. [Do other universities differ?]
- Equipment: Any computer with a supported browser. [Is there a list of supported browsers? Are there any network restrictions?]

## Scenario (cont.)

10. Student ***A*** logs off
11. Later, Student ***A*** log in, finishes the exam, submits the answers, and logs out. [Is this process any different from the initial work on this exam?]
12. Student ***A*** has now completed the exam. The student selects an option that submits the exam to the grading system. [What if the student has not attempted every question? Is the grader or the student notified?]

# Developing a Scenario with a Client

## Example

Requirements are being developed for a system that will enable university students to take exams online from their own rooms using a web browser

- Purpose: Scenario that describes the use of an online Exam system by a representative student
- Individual: [Who is a typical student?] Student **A**, senior at GUC, major in CS. [Do other universities differ?]
- Equipment: Any computer with a supported browser. [Is there a list of supported browsers? Are there any network restrictions?]

## Scenario (cont.)

13. Student $A$ now wishes to change a solution. The system does not permit changes once the solution has been submitted. [Can the student still see the solutions?]
14. Later, Student $A$ logs in to check the grades. [When are grades made available? How does the student know?]
15. Student $A$ requests a regrade. [What are the policies? What are the procedures?]

# Step 2 – Requirements Analysis and Modeling

- Scenarios are useful in discussing a proposed system with a client, but requirements need to be made more precise before a system is fully understood
- This is the purpose of **requirements modeling**
- **A model is a simplification of reality** to better understand system and its complexity

## Modeling Methods

| Notation | Description |
|---|---|
| Natural language | Requirements are written using **numbered sentences in natural language**. Each sentence should express one requirement |
| Structured natural language | Requirements are written in natural language on a **standard form or template**. Each field provides information about an aspect of the requirement. **Best used for embedded control sw** |
| Design description languages | This approach uses **a language like a programming language**, but with more abstract features to specify the requirements by defining an operational model of the system. This approach is now **rarely used** although it can be useful for interface specifications |
| Graphical notations | **Graphical models**, supplemented by text annotations, are used to define the functional requirements for the system; use case and sequence diagrams are commonly used |
| Mathematical specifications | These notations are based on **mathematical concepts** such as finite-state machines or sets. Although these unambiguous specifications can reduce the ambiguity in a requirements document, **most customers don't understand a formal specification**. They cannot check that it represents what they want and are reluctant to accept it as a system contract |

# Natural Language Specification

## Guidelines for Writing Requirements

- Invent a standard format and use it for all requirements
- Use language in a consistent way. Use shall for mandatory requirements, should for desirable requirements
- Use text highlighting to identify key parts of the requirement
- Avoid the use of computer jargon
- Include an explanation (rationale) of why a requirement is necessary

## Problems with Natural Language

- Lack of clarity
  - Precision is difficult without making the document difficult to read
- Requirements confusion
  - Functional and nonfunctional requirements tend to be mixed-up
- Requirements amalgamation
  - Several different requirements may be expressed together

# Structured Specifications

## Form-based Specifications

- Definition of **function** or entity
- Description of **inputs** and where they come from
- Description of **outputs** and where they go to
- Information needed for the **computation** and other entities used
- Description of the **action** to be taken
- Pre and post **conditions** (if appropriate)
- The **side effects** (if any) of the function

## Example

*Insulin Pump/Control Software/SRS/3.3.2*

**Function** Compute insulin dose: safe sugar level.

**Description**

Computes the dose of insulin to be delivered when the current measured sugar level is in the safe zone between 3 and 7 units.

**Inputs** Current sugar reading (r2); the previous two readings (r0 and r1).

**Source** Current sugar reading from sensor. Other readings from memory.

**Outputs** CompDose—the dose in insulin to be delivered.

**Destination** Main control loop.

# Graphical Specifications

High-level **graphical models**, supplemented by **text annotations**

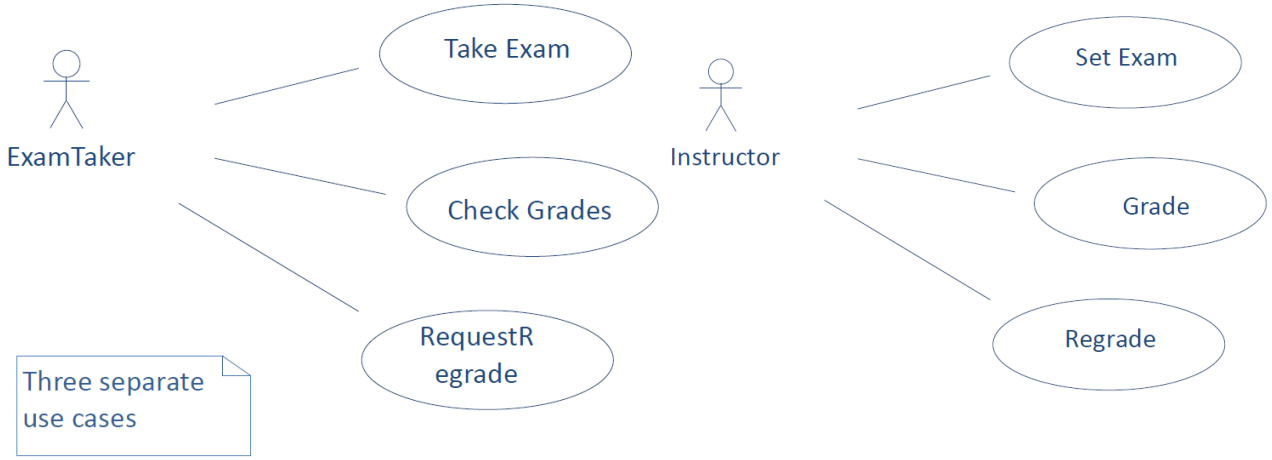
- A diagram is the graphical representation of a set of elements, usually rendered as a connected graph of **vertices** (things) and **arcs** (relationships)
- **Use case diagrams** and **sequence diagrams** are commonly used
- Each diagram is supported by technical documentation that specifies in more detail the model represented by the diagram
- **A diagram without a documented specification is of little value**

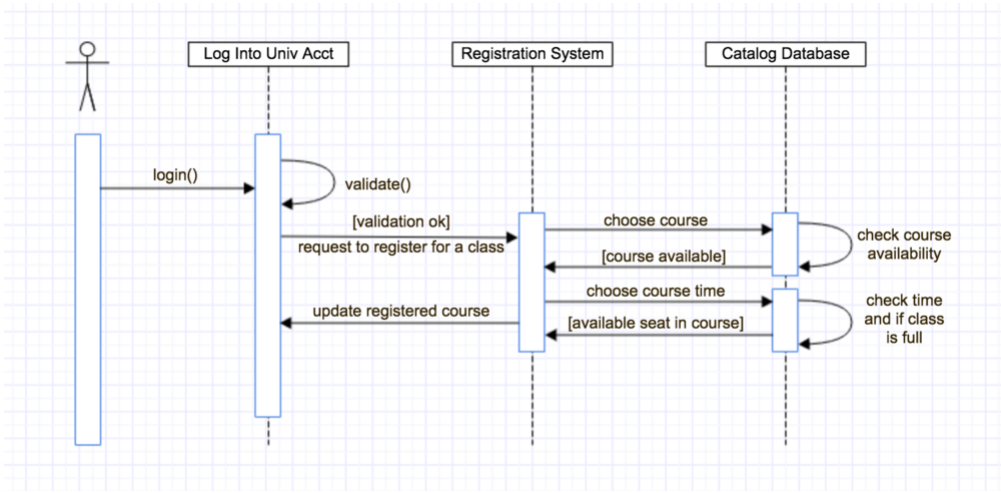
# Graphical Specifications

## Use Cases

- Use-cases are a **scenario modeling method**
- Use-case diagrams identify the `actors` in an interaction and a `use case` which describe the interaction itself
- A **set of use cases should describe all possible interactions** with the system
- Sequence diagrams may be used to add detail to use-cases by showing the sequence of event processing in system

## Example



# Graphical Specifications

## Use Cases

- An **actor** is a user of a system in a particular **role**
- An actor can be human or an external system
- Actor must be a beneficiary of the use case
- **Actor is role, not an individual**
- Choose actor names that describe the role, not generic names, such as "user" or "client"
- A **use case** is a task that an actor needs to perform with the help of the system

## Describing Use Cases

Description/Documentation should include:

- The **name** of the use case, which should summarize its purpose
- The **actor or actors**
- The **flow of events**
- Assumptions about **entry conditions**

# Graphical Specifications

## Describing Use Cases – Example

- **Name of Use Case: Take Exam**
- **Actor(s):** `ExamTaker`
- **Flow of events:**

1. `ExamTaker` connects to the Exam server
2. Exam server checks whether `ExamTaker` is already authenticated and runs authentication process if necessary
3. `ExamTaker` selects an exam from a list of options
4. `ExamTaker` repeatedly selects a question and either types in a solution, attaches a file with a solution, edits a solution or attaches a replacement file
5. `ExamTaker` either submits completed exam or saves current state
6. When a completed exam is submitted, Exam server checks that all questions have been attempted and either sends acknowledgement to `ExamTaker`, or saves current state and notifies `ExamTaker` of incomplete submission
7. `ExamTaker` logs out

- **Entry conditions:**

1. `ExamTaker` must have authentication credentials
2. Computing requirements: supported browser

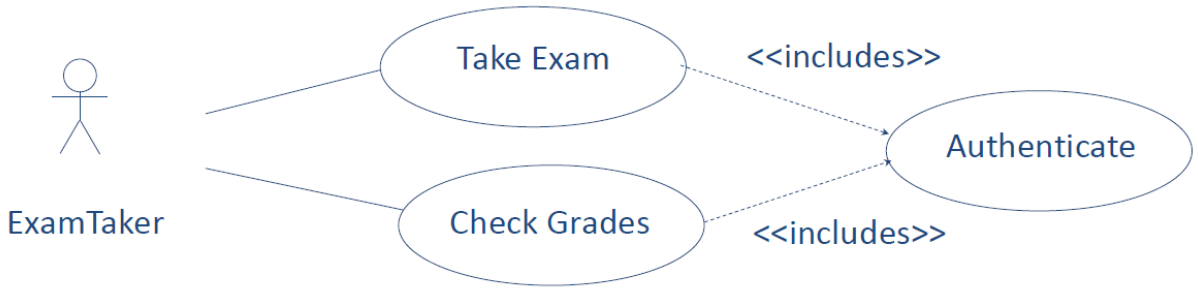
# Graphical Specifications

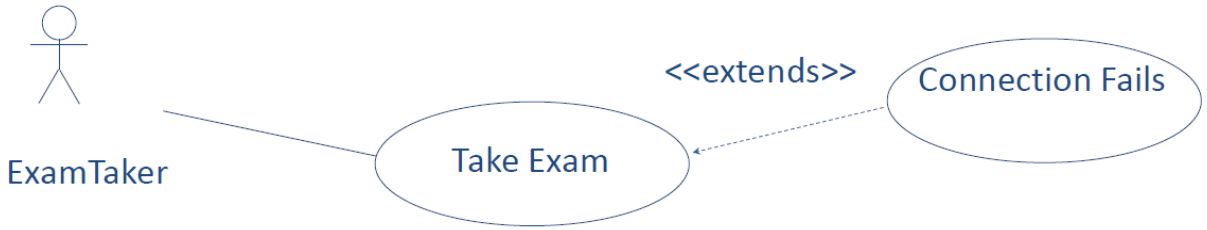
## Relationships Between Use Cases

- `<<includes>>` is used for use cases that are in the flow of events of the main use case

- `<<extends>>` is used for exceptional conditions, especially those that can occur at any time

## Example



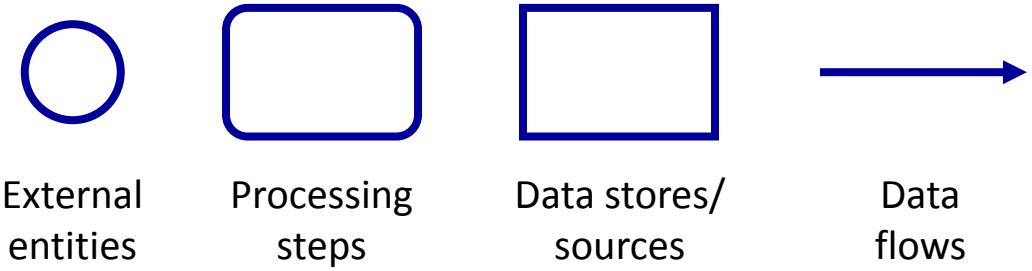



# Graphical Specifications

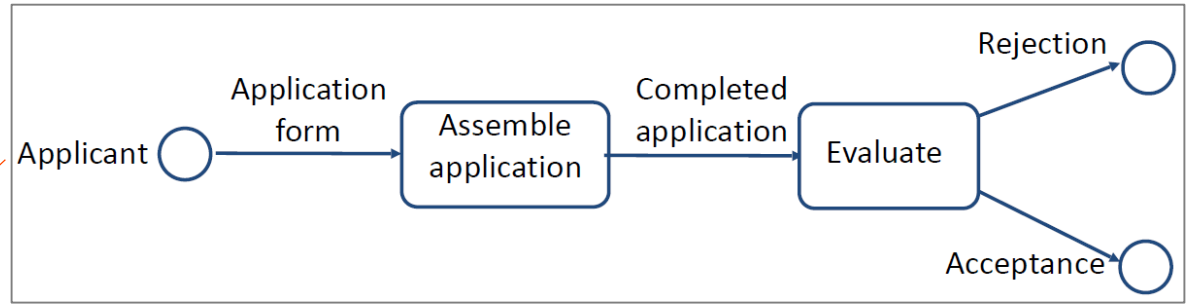
## Data Flow Models

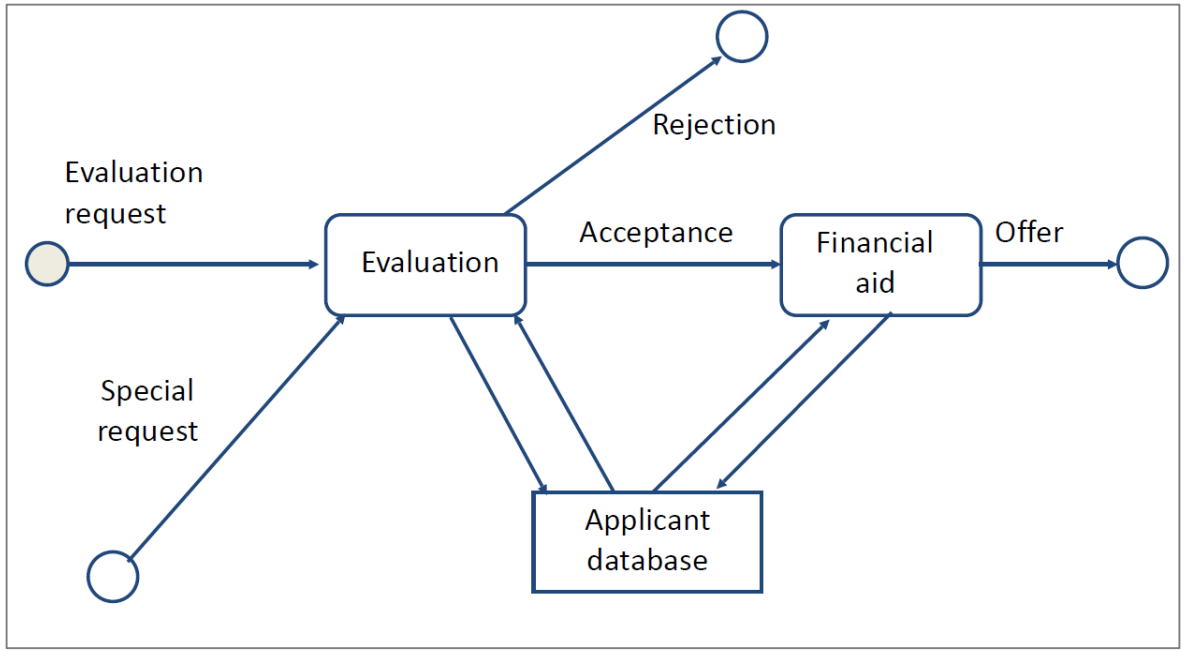
An informal modeling technique to show the flow of data through a system

Where is the data stored? Is there supporting information?




## Example – University Admissions



# Exercise – The Pizza Ordering System

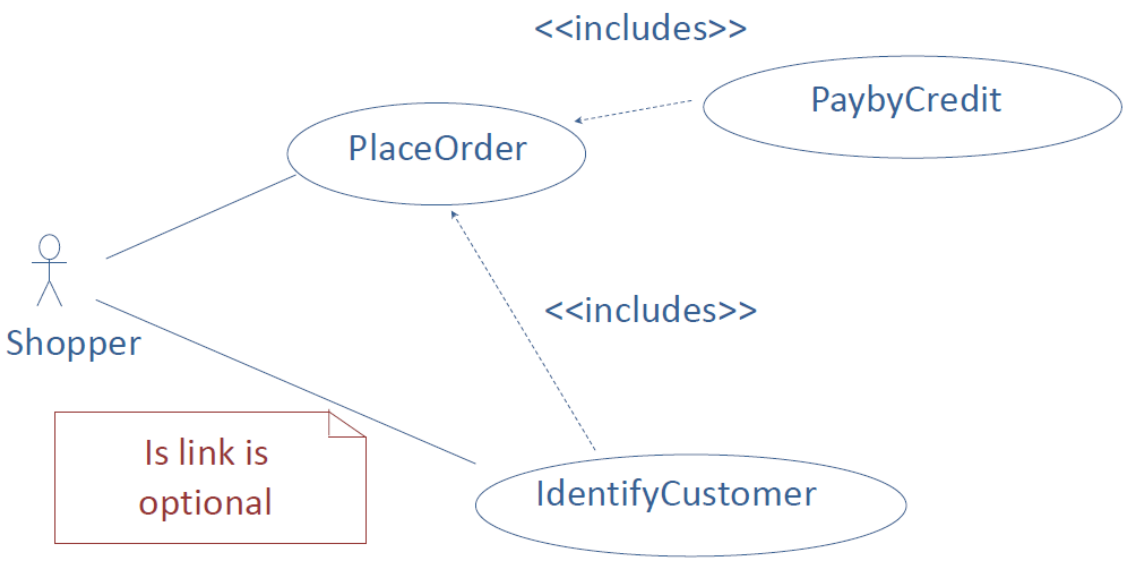
The Pizza Ordering System allows the user of a web browser to order pizza for home delivery. To place an order, a shopper searches to find items to purchase, adds items one at a time to a shopping cart, and possibly searches again for more items.

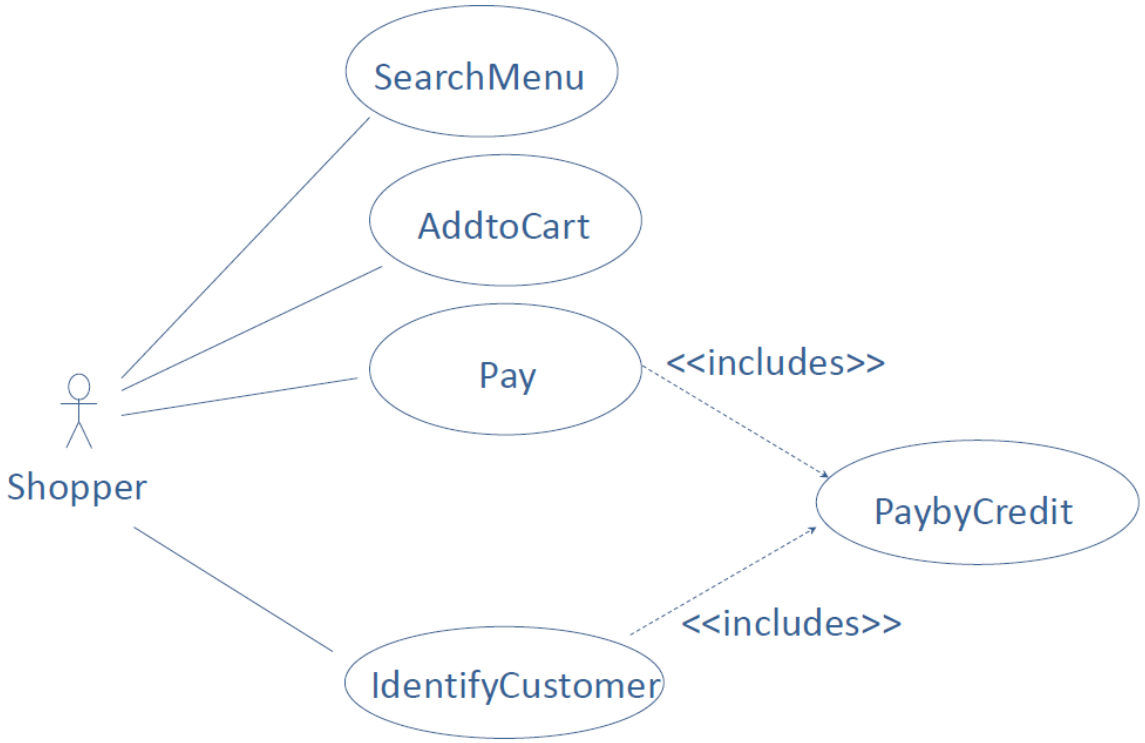
When all items have been chosen, the shopper provides a delivery address. If not paying with cash, the shopper also provides credit card information.

The system has an option for shoppers to register with the pizza shop. They can then save their name and address information, so that they do not have to enter this information every time that they place an order.

Develop a **use case diagram**, for a use case for placing an order, **`PlaceOrder`**. The use case should show a relationship to two previously specified use cases, **`IdentifyCustomer`**, which allows a user to register and log in, and **`PaybyCredit`**, which models credit card payments.

# Which one is correct?



# Characteristics of Good Requirements

Wrong: The system must be easy to use.

Correct: After one day's training, an operator should be able to process 50 transactions per hour.

Wrong: Client wants a report transcript in 5 different formats.

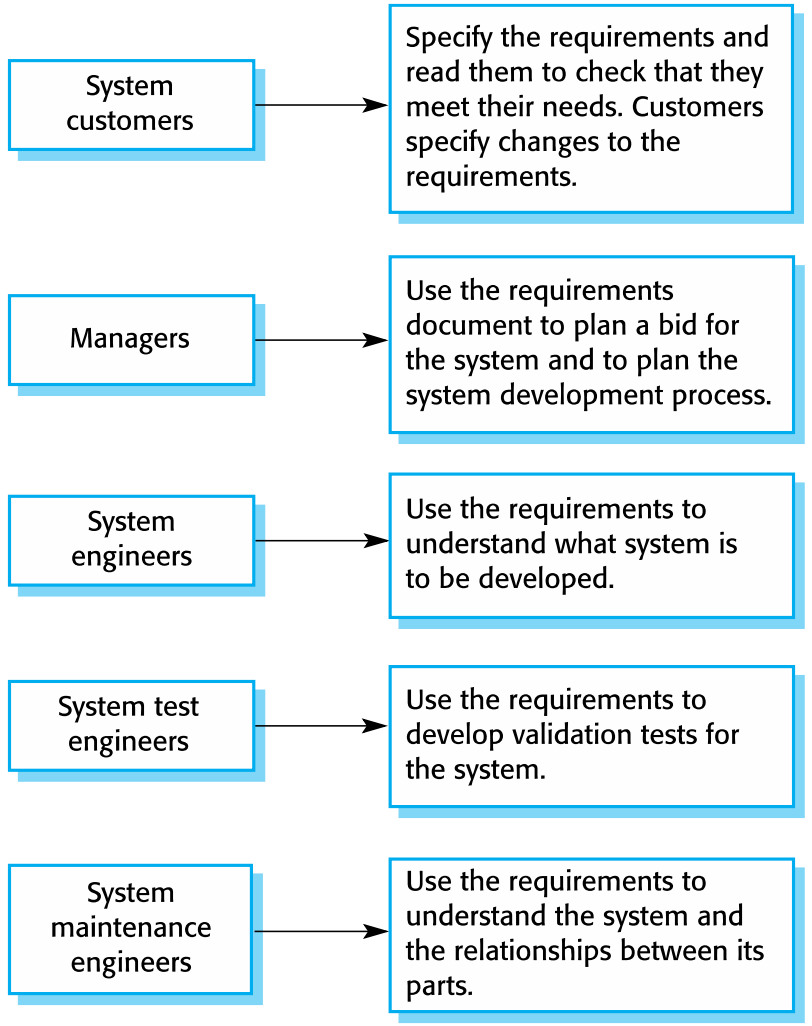
Correct: Negotiate two prominent formats.

Goal of requirements engineering is to:

- **Understand** the requirements **in appropriate detail** (e.g. legal, data flow)
- **Define** requirements that are both **complete** and **consistent** (no conflicts)
- **Define** requirements in a manner that is **clear to the client**. This may be a written specification, prototype system, etc.
- **Define** the requirements in a manner that is **clear to the people** who will **design, implement, and maintain the system**
- **Ensure** that the **client and developers understand** the requirements and their **implications**
- **Negotiate and prioritize** requirements with clients

# Step 3 – Software Requirements Specification

- The **SRS document** is the **official statement of what is required of the system developers**
- Should include both a **definition of user requirements** and a **specification of the system requirements**
- It is **NOT a design document**. It should set WHAT the system should do and not HOW it should do it
- Standards have been designed for SRS document, e.g. IEEE standard. Mostly applicable to requirements for large systems engineering projects




https://freshcodeit.com/freshcode-post/creating-srs-step-by-step-by-analyzing-requirements

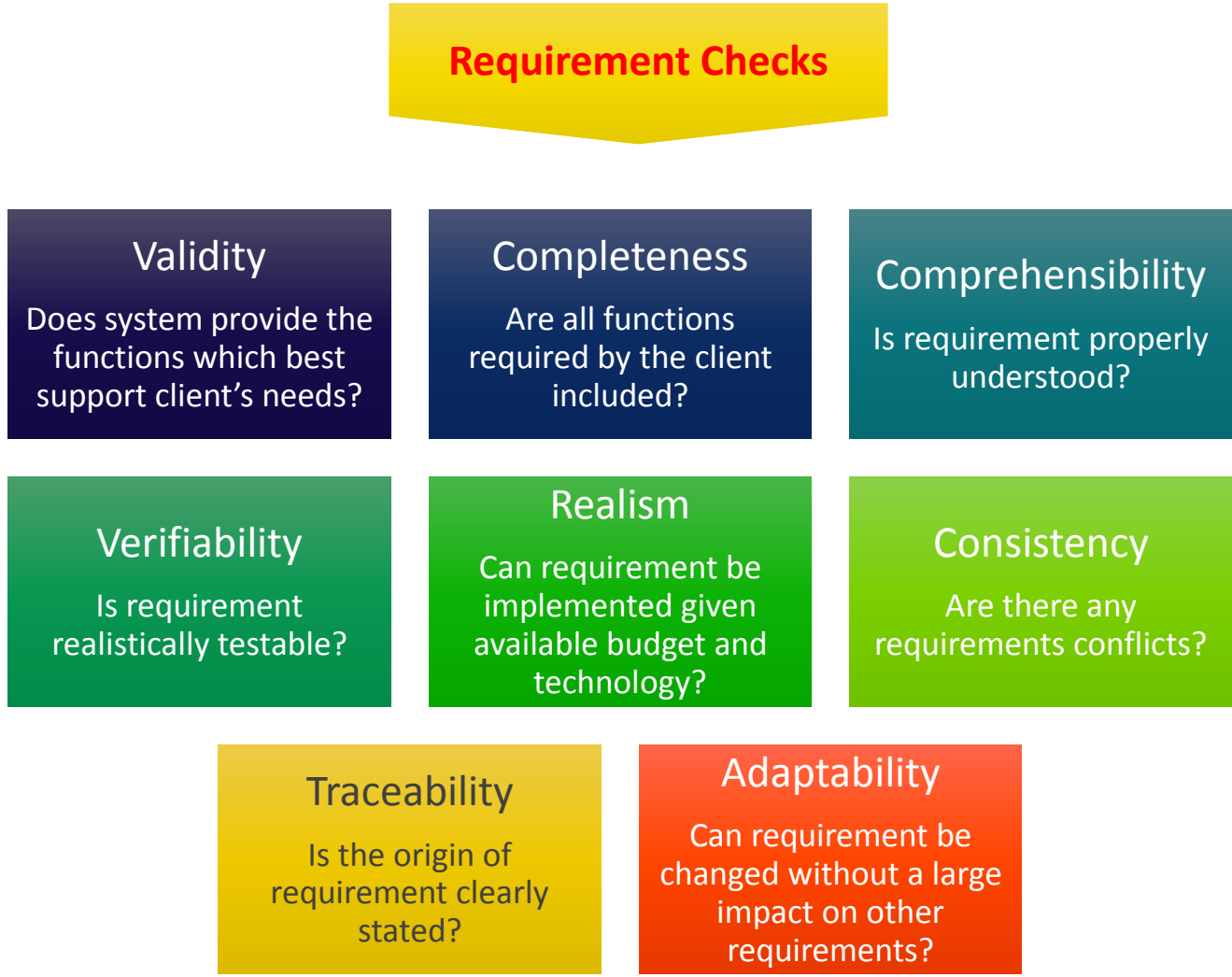
# Step 4 – Requirements Validation

- Once you have the SRS, you need to verify and validate the requirements
- Demonstrate that the **requirements define the system that the customer really wants**
- **Requirements error costs are high** so **validation is very important**
  - Fixing a requirements error after delivery may cost up to 100 times the cost of fixing an implementation error

- **Good communications** between developers, customers and users **can resolve problems at an early stage**
- **Regular reviews** should be held while the requirements definition is being formulated
- **Both client and contractor staff should be involved in reviews**
- Reviews may be **formal or informal**

# Requirements Validation Techniques

- **Requirements reviews**
  - Systematic **manual analysis** of the requirements
- **Prototyping**
  - Using an **executable model** of the system to check requirements
- **Test case generation**
  - Developing **tests for requirements** to check testability



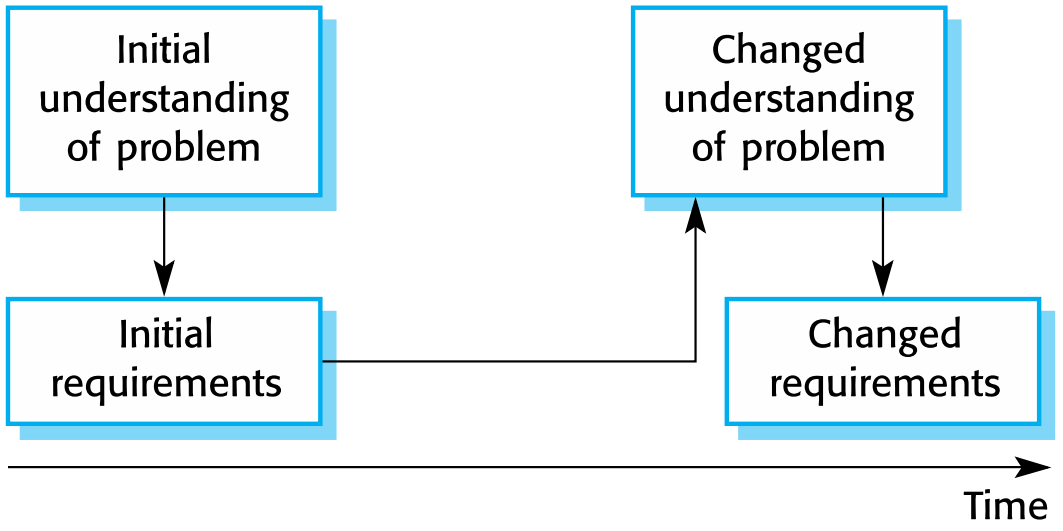
# Requirements May Evolve Over Time

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

BUT …

**Clients may have only a partial understanding of requirements**

- 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



# Requirements May Evolve Over Time

**New requirements emerge as a system is being developed and after it has gone into use**

- Business and technical environment of the system always changes after installation
- New hardware may be introduced
- It may be necessary to interface the system with other systems
- Business priorities may change (with consequent changes in the system)
- New legislation and regulations may be introduced that the system must necessarily abide by
- Requirements of system customers because of organizational and budgetary constraints may conflict with end-user requirements
- Large systems usually have a diverse user community, with many users having different requirements and priorities that may be conflicting or contradictory

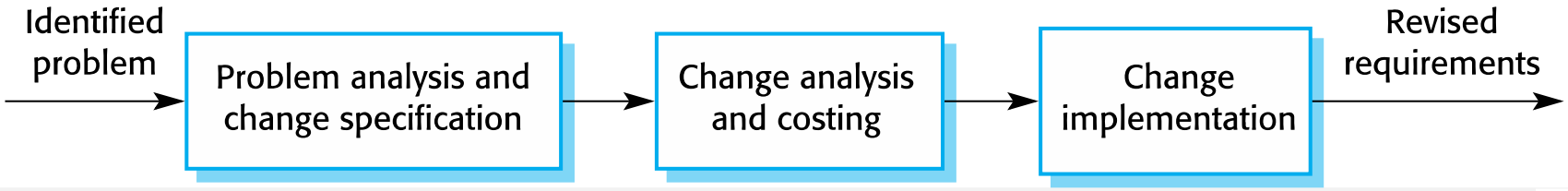
# Requirements Change Management

- **Requirements management** is the process of **managing changing requirements during the requirements engineering process and system development**
- You need to **keep track of individual requirements** (e.g. via unique identifiers) and **maintain links between dependent requirements** so that you can **assess the impact of requirements changes**
- You need to devise **traceability policies** that define the relationships between requirements and between the requirements and the system design
- You need to establish a **formal process for making change proposals** and linking these to system requirements

# Requirements Change Management

- **Change management process** is the activities that assess impact and cost of changes

1. Problem analysis and change specification
   - Problem or change proposal is analyzed to check that it is valid. This analysis is fed back to the change requestor who may respond with a more specific requirements change proposal, or decide to withdraw the request
2. Change analysis and costing
   - Effect of the proposed change is assessed using traceability information and general knowledge of the system requirements. A decision is made whether or not to proceed with the change
3. Change implementation
   - The requirements document and, where necessary, the system design and implementation, are modified. Ideally, the document should be organized so that changes can be easily implemented






# NEXT WEEK on SE

# Disclaimer

Content is adapted from Ian Sommerville's book slides, Ivan Marsic's lecture slides at Rutgers University, and William Y. Arms' lecture slides from Cornell University

# Thank You

mervat.abuelkheir@guc.edu.eg