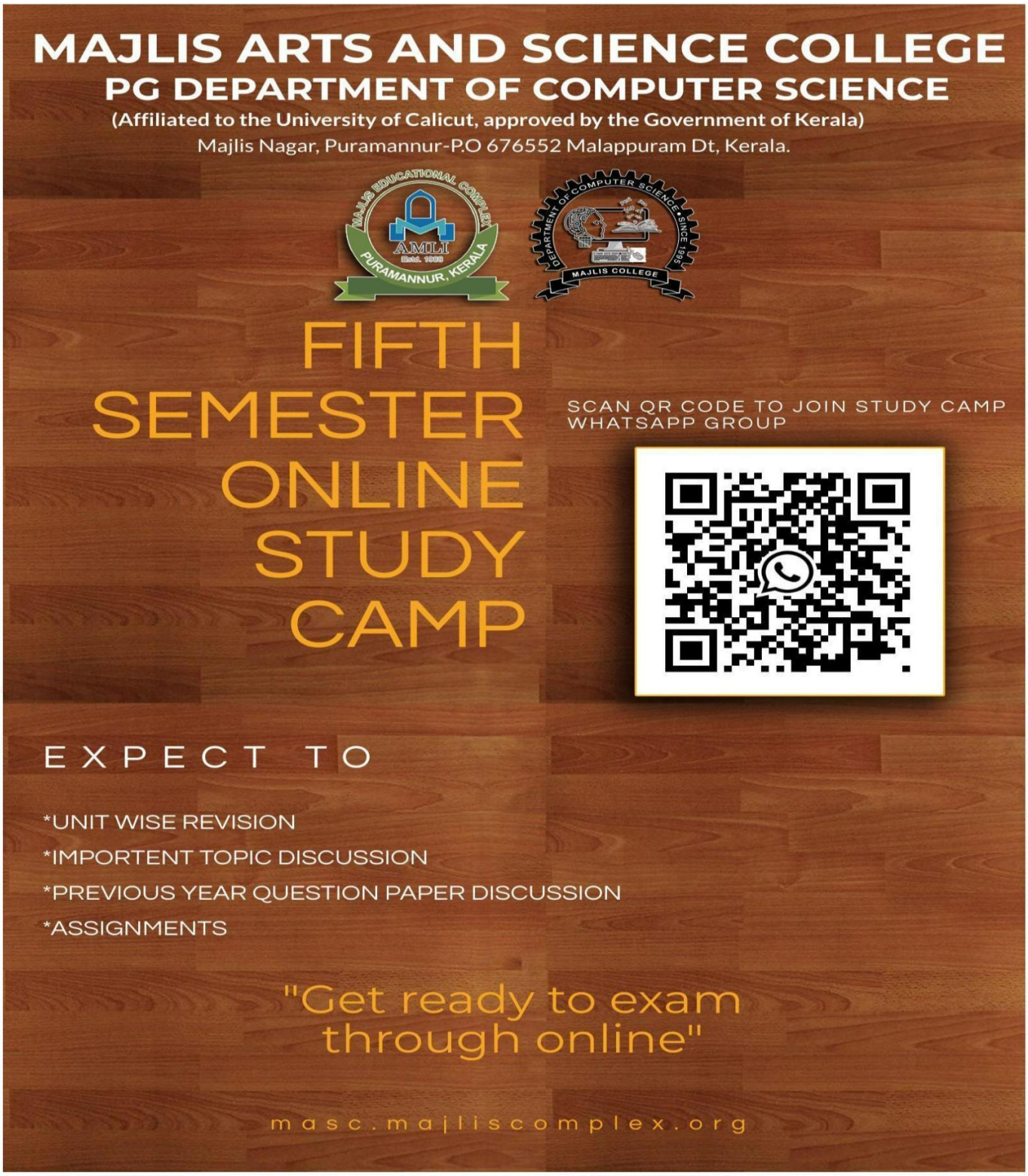


SOFTWARE ENGINEERING

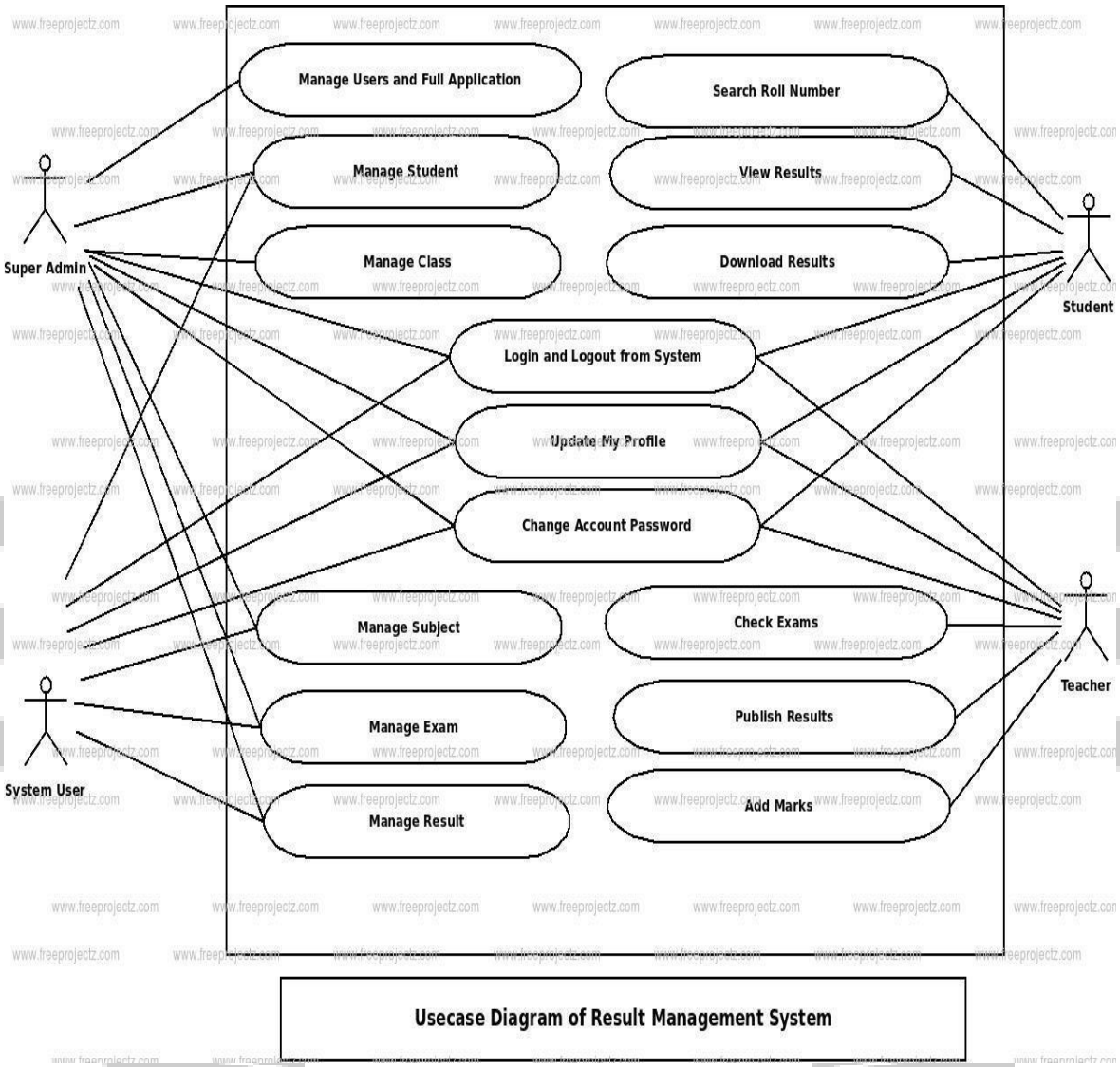
MODULE 2

1. In the classical waterfall model during _____ phase is the Software Requirement Specification (SRS) document produced ?

   Ans: Second phase

2. Draw Use Case Dingram of Result Management System of M.Tech. Programme.

Ans:



Usecase Diagram of Result Management System

3. What are the components of state transition diagram ? Give example.

Ans:

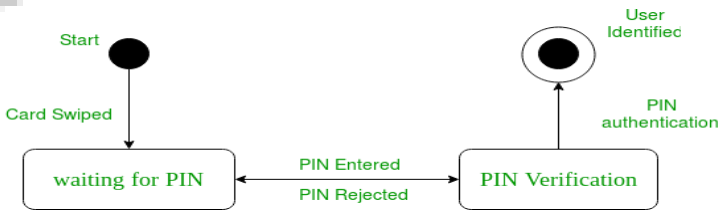


Figure – a state diagram for user verification

The basic purpose of a **state diagram** is to portray various changes in state of the class and not the processes or commands causing the changes.

Basic components of a statechart diagram –

Initial state – We use a black filled circle represent the initial state of a System or a class.

Figure – initial state notation

Transition – We use a solid arrow to represent the transition or change of control from one state to another. The arrow is labelled with the event which causes the change in state.

Figure – transition

State – We use a rounded rectangle to represent a state. A state represents the conditions or circumstances of an object of a class at an instant of time.

Figure – state notation

Fork – We use a rounded solid rectangular bar to represent a Fork notation with incoming arrow from the parent state and outgoing arrows towards the newly created states. We use the fork notation to represent a state splitting into two or more concurrent states.

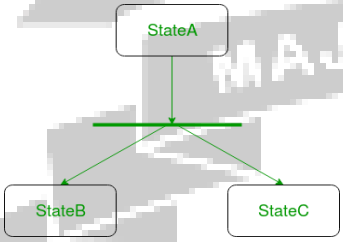


Figure – a diagram using the fork notation

Join – We use a rounded solid rectangular bar to represent a Join notation with incoming arrows from the joining states and outgoing arrow towards the common goal state. We use the join notation when two or more states concurrently converge into one on the occurrence of an event or events.

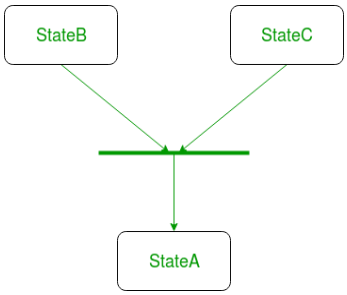


Figure – a diagram using join notation

Self transition – We use a solid arrow pointing back to the state itself to represent a self transition. There might be scenarios when the state of the object does not change upon the occurrence of an event. We use self transitions to represent such cases.

Figure – self transition notation

Composite state – We use a rounded rectangle to represent a composite state also.We represent a state with internal activities using a composite state.

Figure – a state with internal activities

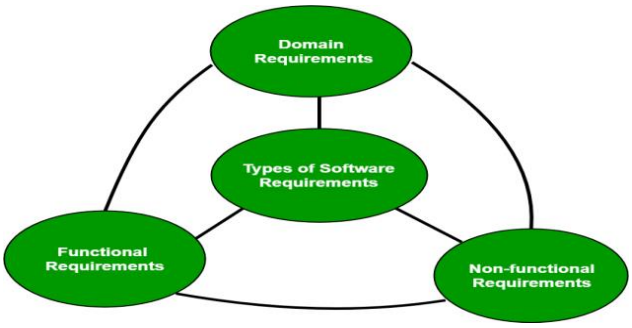
Final state – We use a filled circle within a circle notation to represent the final state in a state machine diagram.

Figure – final state notation

4. What are the different types of requirements ? Explain in detail.

Ans:A software requirement can be of 3 types:

- Functional requirements
- Non-functional requirements
- Domain requirements



Functional Requirements: These are the requirements that the end user specifically demands as basic facilities that the system should offer. All these functionalities need to be necessarily incorporated into the system as a part of the contract. These are represented or stated in the form of input to be given to the system, the operation performed and the output expected. They are basically the requirements stated by the user which one can see directly in the final product, unlike the non-functional requirements. For example, in a hospital management system, a doctor should be able to retrieve the information of his patients. Each high-level functional requirement may involve several interactions or dialogues between the system and the outside world. In order to accurately describe the functional requirements, all scenarios must be enumerated.

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.

They basically deal with issues like:

- Portability
- Security
- Maintainability
- Reliability
- Scalability
- Performance
- Reusability
- Flexibility

The process of specifying non-functional requirements requires the knowledge of the functionality of the system, as well as the knowledge of the context within which the system will operate.

Domain requirements: Domain requirements are the requirements which are characteristic of a particular category or domain of projects. The basic functions that a system of a specific domain must necessarily exhibit come under this category. For instance, in an academic software that maintains records of a school or college, the functionality of being able to access the list of faculty and list of students of each grade is a domain requirement. These requirements are therefore identified from that domain model and are not user specific.

5. What is the first step of requirement elicitation?

Ans: Identifying Stakeholder

6. What are the Objectives of Requirement Analysis ?

Ans: Requirement Analysis, also known as Requirement Engineering, is the process of defining user expectations for a new software being built or modified. In software engineering, it is sometimes referred to loosely by names such as requirements gathering or requirements capturing.

Objectives of Requirement Analysis

Identify customer's needs.

• Evaluate system for feasibility.

• Perform economic and technical analysis.

• Allocate functions to system elements.

• Establish schedule and constraints.

• Create system definitions.

The various steps of requirement analysis are

(i) Draw the context diagram: The context diagram is a simple model that defines the boundaries and interfaces of the proposed systems with the external world. It identifies the entities outside the proposed system that interact with the system.

(ii) Development of a Prototype (optional): One effective way to find out what the customer wants is to construct a prototype, something that looks and preferably acts as part of the system they say they want.

We can use their feedback to modify the prototype until the customer is satisfied continuously. Hence, the prototype helps the client to visualize the proposed system and increase the understanding of the requirements. When developers and users are not sure about some of the elements, a prototype may help both the parties to take a final decision.

Some projects are developed for the general market. In such cases, the prototype should be shown to some representative sample of the population of potential purchasers. Even though a person who tries out a prototype may not buy the final system, but their feedback may allow us to make the product more attractive to others.

The prototype should be built quickly and at a relatively low cost. Hence it will always have limitations and would not be acceptable in the final system. This is an optional activity.

(iii) Model the requirements: This process usually consists of various graphical representations of the functions, data entities, external entities, and the relationships between

them. The graphical view may help to find incorrect, inconsistent, missing, and superfluous requirements. Such models include the Data Flow diagram, Entity-Relationship diagram, Data Dictionaries, State-transition diagrams, etc.

(iv) Finalise the requirements: After modeling the requirements, we will have a better understanding of the system behavior. The inconsistencies and ambiguities have been identified and corrected. The flow of data amongst various modules has been analyzed. Elicitation and analyze activities have provided better insight into the system. Now we finalize the analyzed requirements, and the next step is to document these requirements in a prescribed format.

7. What kinds of errors are sought out during requirements validation? Explain.

Ans: • May be requirements are not consistent with the overall objectives for the System/product
• Requirements may not be specified at the proper level of abstraction. That is, do some requirements provide a level of technical detail that is inappropriate at this stage?
• Requirements that are unnecessary or does it represent an add-on feature that may not be essential to the objective of the system.
• Unbounded and ambiguous requirements.
• Requirements may conflict with other requirements.
• There may be unachievable requirements.
• Requirements that are not testable are considered as errors in validation.

8. Explain seven distinct requirements engineering functions.

Ans: The broad spectrum of tasks and techniques that lead to an understanding of requirements
is called requirements engineering. It encompasses seven distinct tasks: inception, elicitation, elaboration, negotiation, specification, validation, and management. It is important to note that
some of these tasks occur in parallel and all are adapted to the needs of the project.

1. Inception

- Inception is a task where the requirement engineering asks a set of questions to establish a software process.
- In this task, it understands the problem and evaluates with the proper solution.
- It collaborates with the relationship between the customer and the developer.
- The developer and customer decide the overall scope and the nature of the question.

2. Elicitation

Elicitation means to find the requirements from anybody.
The requirements are difficult because the **following problems occur in elicitation**.

Problem of scope: The customer give the unnecessary technical detail rather than clarity of the overall system objective.

Problem of understanding: Poor understanding between the customer and the developer regarding various aspect of the project like capability, limitation of the computing environment.

Problem of volatility: In this problem, the requirements change from time to time and it is difficult while developing the project.

3. Elaboration

- In this task, the information taken from user during inception and elaboration and are expanded and refined in elaboration.
- Its main task is developing pure model of software using functions, feature and constraints of a software.

4. Negotiation

- In negotiation task, a software engineer decides the how will the project be achieved with limited business resources.
- To create rough guesses of development and access the impact of the requirement on the project cost and delivery time.

5. Specification

- In this task, the requirement engineer constructs a final work product.
- The work product is in the form of software requirement specification.
- In this task, formalize the requirement of the proposed software such as informative, functional and behavioral.
- The requirements are formalize in both graphical and textual formats.

6. Validation

- The work product is built as an output of the requirement engineering and that is accessed for the quality through a validation step.
- The formal technical reviews from the software engineer, customer and other stakeholders helps for the primary requirements validation mechanism.

7. Requirement management

- It is a set of activities that help the project team to identify, control and track the requirements and changes can be made to the requirements at any time of the ongoing project.
- These tasks start with the identification and assign a unique identifier to each of the requirement.
- After finalizing the requirement traceability table is developed.
- The examples of traceability table are the features, sources, dependencies, subsystems and interface of the requirement.

9. How will you negotiate the requirements?

Ans: In an ideal requirements engineering context, the inception, elicitation, and elaboration tasks determine customer requirements in sufficient detail to proceed to subsequent Software engineering activities. Unfortunately, this rarely happens. In reality, You may have to enter into a negotiation with one or more stakeholders. In most cases, stakeholders are asked to balance functionality, performance, and other product or system characteristics against cost and time-to-market. The intent of this negotiation is to develop a project plan that meets stakeholder needs while at the same time reflecting the real-world constraints (e.g., time, people, budget) that have been placed on the software team.

The best negotiations strive for a "win-win" result. That is, stakeholders win by getting the system or product that satisfies the majority of their needs and you (as a member of the software team) win by working to realistic and achievable budgets and deadlines.

There are a set of negotiation activities at the beginning of each software process iteration. Rather than a single customer communication activity, the following activities are defined:

1. Identification of the system or subsystem's key stakeholders.
2. Determination of the stakeholders' "win conditions."
3. Negotiation of the stakeholders' win conditions to reconcile them into a set of win-win conditions for all concerned (including the software team).

Successful completion of these initial steps achieves a win-win result, which becomes the key criterion for proceeding to subsequent software engineering activities.

10. What do you mean by Quality Function Deployment (QFD) ?

Ans: Quality function deployment (QFD) is a quality management technique that translates the needs of the customer into technical requirements for software. QFD "concentrates on maximizing customer satisfaction from the software engineering process".

QFD identifies three types of requirements:

Normal requirements. The objectives and goals that are stated for a product or system during meetings with the customer. If these requirements are present, the customer is satisfied. Examples of normal requirements might be requested types of graphical displays, specific system functions, and defined levels of performance.

Expected requirements. These requirements are implicit to the product or system and may be so fundamental that the customer does not explicitly state them. Their absence will be a cause for significant dissatisfaction. Examples of expected requirements are: ease of human/machine interaction, overall operational correctness and reliability, and ease of software installation.

Exciting requirements. These features go beyond the customer's expectations and prove to be very satisfying when present. For example, software for a new mobile phone comes with standard features, but is coupled with a set of unexpected capabilities (e.g., multitouch screen, visual voice mail) that delight every user of the product.

QFD uses customer interviews and observation, surveys, and examination of historical data (e.g., problem reports) as raw data for the requirements gathering activity. These data are then translated into a table of requirements—called the customer voice table—that is reviewed with the customer and other stakeholders.

11. How the requirements model is built?

Ans: The intent of the analysis model is to provide a description of the required informational,
functional, and behavioral domains for a computer-based system. The model changes dynamically as you learn more about the system to be built, and other stakeholders understand more about what they really require. As the requirements model evolves, certain elements will become relatively stable, providing a solid foundation for the design tasks that follow.

Elements of the Requirements Model

- Scenario-based elements.
  The system is described from the user's point of view using a scenario-based approach. For example, basic use cases and their corresponding use-case diagrams evolve into more elaborate template-based use cases. Scenario-based elements of the requirements model are often the first part of the model that is developed. As such, they serve as input for the creation of other modeling elements.
- Class-based elements.
  Each usage scenario implies a set of objects that are manipulated as an actor interacts with the system. These objects are categorized into classes—a collection of things that have similar attributes andcommonbehaviors

- Behavioral elements.
  The behavior of a computer-based system can have a profound effect on the design that is chosen and the implementation approach that is applied. Therefore, the requirements model must provide modeling elements that depictbehavior.The state diagram is one method for representing the behavior of a system by depicting its states and the events that cause the system to change state. A state is any externally observable mode of behavior.
- Flow-oriented elements.
  Information is transformed as it flows through a computer-based system. The system accepts input in a variety of forms, applies functions to transform it, and produces output in a variety of forms. Input may be a control signal transmitted by a transducer, a series of numbers typed by a human operator, a packet of information transmitted on a network link, or a voluminous data file retrieved from secondary storage. The transform(s) may comprise a single logical comparison, a complex numerical algorithm, or a rule-inference approach of an expert system. Output may light a single LED or produce a 200-page report.

12. How will you elicit the requirments. Discuss

Ans: Requirements elicitation (also called requirements gathering) combines elements of problem solving, elaboration, negotiation, and specification. In order to encourage

a collaborative, team-oriented approach to requirements gathering, stakeholders work together to identify the problem, propose elements of the solution, negotiate different approaches and specify a preliminary set of solution requirements.

There are various ways to discover requirements

Interviews

Interviews are strong medium to collect requirements. Organization may conduct several types of interviews such as:

- Structured (closed) interviews, where every single information to gather is decided in advance, they follow pattern and matter of discussion firmly.
- Non-structured (open) interviews, where information to gather is not decided in advance, more flexible and less biased.
- Oral interviews
- Written interviews
- One-to-one interviews which are held between two persons across the table.
- Group interviews which are held between groups of participants. They help to uncover any missing requirement as numerous people are involved.

Surveys

Organization may conduct surveys among various stakeholders by querying about their expectation and requirements from the upcoming system.

Questionnaires

A document with pre-defined set of objective questions and respective options is handed over to all stakeholders to answer, which are collected and compiled.

A shortcoming of this technique is, if an option for some issue is not mentioned in the questionnaire, the issue might be left unattended.

Task analysis

Team of engineers and developers may analyze the operation for which the new system is required. If the client already has some software to perform certain operation, it is studied and requirements of proposed system are collected.

Domain Analysis

Every software falls into some domain category. The expert people in the domain can be a great help to analyze general and specific requirements.

Brainstorming

An informal debate is held among various stakeholders and all their inputs are recorded for further requirements analysis.

Prototyping

Prototyping is building user interface without adding detail functionality for user to interpret the features of intended software product. It helps giving better idea of requirements. If there is no software installed at client's end for developer's reference and the client is not aware of its own requirements, the developer creates a prototype based on initially mentioned requirements. The prototype is shown to the client and the feedback is noted. The client feedback serves as an input for requirement gathering.

Observation

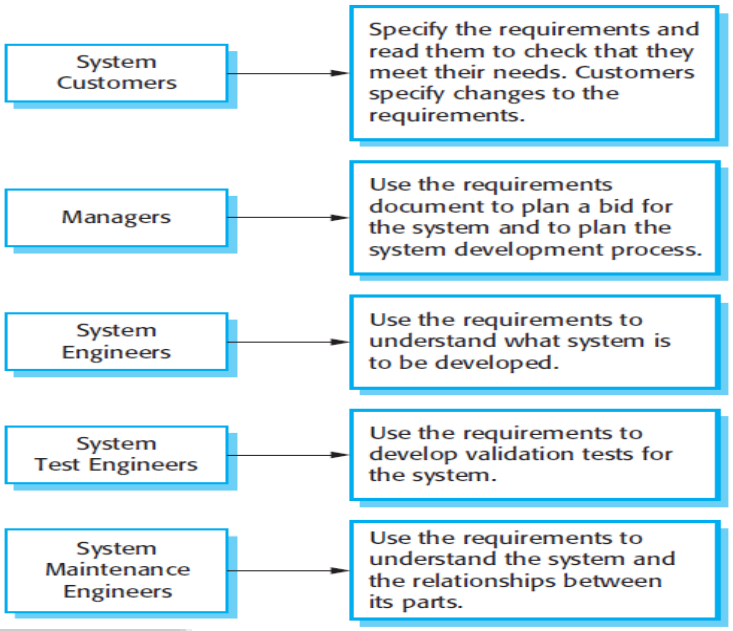
Team of experts visit the client's organization or workplace. They observe the actual working of the existing installed systems. They observe the workflow at client's end and how execution problems are dealt. The team itself draws some conclusions which aid to form requirements expected from the software.

13. The _____ is the final outcome of the requirements analysis and specification phase.

Ans: SRS Document

14. What are the different categories of users of SRS document and what are the different categories of customer requirement ?

Ans:



For customer requirements refer question 4.

15. Explain requirement gathering techniques.

Ans: There are a number of requirements elicitation methods. Few of them are listed below –

1. Interviews
2. Brainstorming Sessions

3. Facilitated Application Specification Technique (FAST)
4. Quality Function Deployment (QFD)
5. Use Case Approach

The success of an elicitation technique used depends on the maturity of the analyst, developers, users, and the customer involved.

1.Interviews:

Objective of conducting an interview is to understand the customer's expectations from the software.

It is impossible to interview every stakeholder hence representatives from groups are selected based on their expertise and credibility.

Interviews maybe be open ended or structured.

1. In open ended interviews there is no pre-set agenda. Context free questions may be asked to understand the problem.
2. In structured interview, agenda of fairly open questions is prepared. Sometimes a proper questionnaire is designed for the interview.

2. Brainstorming Sessions:

- It is a group technique
- It is intended to generate lots of new ideas hence providing a platform to share views
- A highly trained facilitator is required to handle group bias and group conflicts.
- Every idea is documented so that everyone can see it.
- Finally a document is prepared which consists of the list of requirements and their priority if possible.

3.FacilitatedApplicationSpecificationTechnique:

It's objective is to bridge the expectation gap – difference between what the developers think they are supposed to build and what customers think they are going to get.

A team oriented approach is developed for requirements gathering. Each attendee is asked to make a list of objects that are-

1. Part of the environment that surrounds the system
2. Produced by the system
3. Used by the system

Each participant prepares his/her list, different lists are then combined, redundant entries are eliminated, team is divided into smaller sub-teams to develop mini-specifications and finally a draft of specifications is written down using all the inputs from the meeting.

4.QualityFunctionDeployment:

In this technique customer satisfaction is of prime concern, hence it emphasizes on the

requirements which are valuable to the customer.
3 types of requirements are identified –

- **Normal requirements –** In this the objective and goals of the proposed software are discussed with the customer. Example – normal requirements for a result management system may be entry of marks, calculation of results, etc
- **Expected requirements –** These requirements are so obvious that the customer need not explicitly state them. Example – protection from unauthorized access.
- **Exciting requirements –** It includes features that are beyond customer's expectations and prove to be very satisfying when present. Example – when unauthorized access is detected, it should backup and shutdown all processes.

The major steps involved in this procedure are –

1. Identify all the stakeholders, eg. Users, developers, customers etc
2. List out all requirements from customer.
3. A value indicating degree of importance is assigned to each requirement.
4. In the end the final list of requirements is categorized as –
    - It is possible to achieve
    - It should be deferred and the reason for it
    - It is impossible to achieve and should be dropped off

5.UseCaseApproach:

This technique combines text and pictures to provide a better understanding of the requirements.
The use cases describe the 'what', of a system and not 'how'. Hence they only give a functional view of the system.

The components of the use case design includes three major things – Actor, Use cases, use case diagram.

1. **Actor –** It is the external agent that lies outside the system but interacts with it in some way. An actor maybe a person, machine etc. It is represented as a stick figure. Actors can be primary actors or secondary actors.
    - Primary actors – It requires assistance from the system to achieve a goal.
    - Secondary actor – It is an actor from which the system needs assistance.
2. **Use cases –** They describe the sequence of interactions between actors and the system. They capture who(actors) do what(interaction) with the system. A complete set of use cases specifies all possible ways to use the system.
3. **Use case diagram –** A use case diagram graphically represents what happens when an actor interacts with a system. It captures the functional aspect of the system.
    - A stick figure is used to represent an actor.

- An oval is used to represent a use case.
- A line is used to represent a relationship between an actor and a use case.

16. Which is not a requirementa elicitation technique ?

(a) Interviews. (b) The use case approach.m

(c) FAST. (d) Data flow diagram.

Ans: (d) Data flow diagram

17. Outcome of requirement specification is:

(a) Design document, (b) Software requirement specification.

(c) Text document (d) None of these.

Ans: (b) Software requirement specification.

18. What is SRS? Explain the need for SRS.

Ans: The production of the requirements stage of the software development process is**Software Requirements Specifications (SRS)** (also called a **requirements document**). This report lays a foundation for software engineering activities and is constructing when entire requirements are elicited and analyzed. **SRS** is a formal report, which acts as a representation of software that enables the customers to review whether it (SRS) is according to their requirements. Also, it comprises user requirements for a system as well as detailed specifications of the system requirements.

SRS serves as an input our other all documents created in later stages of software development life cycle.

- It is a feedback to the customer.
- It's modularized the problem statement.
- It the bases of system design.
- It defines product scope.

19. What is the importance of requirement analysis ? What are the problems in requirement that the analyst needs to identify ?

Ans: Requirements analysis results in the specification of software's operational characteristics,indicates software's interface with other system elements, and establishes constraints that software must meet. Requirements analysis allows you (regardless of whether you're called a software engineer, an analyst, or a modeler) to elaborate on basic requirements established during the inception, elicitation, and negotiation tasks that are part of requirements engineering.Requirements analysis can be a long and tiring process during which many delicate psychological skills are involved. New systems change the environment and relationships between people, so it is important to identify all the stakeholders, take into account all their needs and ensure they understand the implications of the new systems. Analysts can employ several techniques to elicit the requirements from the customer. These may include the development of scenarios (represented as user stories in agile methods), the identification of use cases, the use of workplace observation or ethnography, holding interviews, or focus groups (more aptly named in this context as requirements workshops, or requirements review sessions) and creating requirements lists. Prototyping may be used to develop an example system that can be demonstrated to stakeholders. Where necessary, the analyst will employ a combination of these methods to establish the exact requirements of the stakeholders, so that a system that meets the business needs is produced.

Problem of scope: The customer give the unnecessary technical detail rather than clarity of the overall system objective.

Problem of understanding: Poor understanding between the customer and the developer regarding various aspect of the project like capability, limitation of the computing environment.

Problem of volatility: In this problem, the requirements change from time to time and it is difficult while developing the project.