Assignment 1

① What is the importance of software process models in software engineering.

Software process models provides a structural and step-by-step development process for the software. The reasons for requiring software process models in software engineering are given below

① Organization and structure: Software process models provides a framework for organizing and structuring the software development process. They outline the sequence of activities and tasks that need to be performed, ensuring that development efforts are coordinated and systematic.

② Predictability & control: Process models allows for better predictability & control over the software development process. By following a defined process, project managers can estimate project timelines, allocate resources efficiently and moniter progress more accurately.

③ Quality Assurance: Process models include quality assurance activities that ensure the software meets specified requirements & quality standards. They define review, testing & verification procedures to identify and rectify defects early in the development life cycle, reducing the likelihood of errors & improving overall software quality.

④ Risk Management: Process models assist in identifying and managing risks throughout the software development process. They provide guidelin for risk assessment, mitigation and contingency planning, enabling project teams to to proacti vely address potential issues and minimize their impact on the project.

⑤ Reusability: Process models promote reusabilit by defining reusable components, best practices, and standardized procedures. This allows organi zations to leverage previous successful experien and lessons learned.

Describe the agile development methodology and its advantages over traditional software development approaches.

Agile software engineering combines a philosophy and a set of development guidelines. The philosophy encourages customer satisfaction and early incremental delivery of software, small highly motivated project teams; informal methods; minimal software engineering work products and overall development simplicity.

Software engineers and other project stakeholders work together on an agile team a team that is self-organizing and in control of its own destiny. An agile team fosters communication and collaboration among all who serve on it.

Advantages of Agile development over traditional:

1) Flexibility and adaptability: Agile methodologies, such as scrum embrace change and allow for flexibility throughout the development process. Agile teams can responds quickly to changing requirements, new priorities and customer feedback, enabling them to deliver valuable software in a timely manner.

(2) Iterative and Incremental Development: Agile development emphasizes iterative and incremental development cycles. Rather than waiting until the end of the project to deliver a final product, agile teams work in short iterations, delivering working software at the end of each iteration.

(3) Customer Collaboration: Agile methodologies emphasize continuous customer collaboration throughout the development process. By involving the customer or product owner in regular feedback sessions, agile teams gain a better understanding of the customer's needs and can make necessary adjustments. This collaborative approach ensures that the end product aligns with customer expectations.

(4) Quick delivery: Agile development promotes faster time to market by focusing on delivering working software in short iterations. The iterative approach allows teams to release new features or improvements incrementally, enabling organizations to gain a competitive edge by responding to maket demands more rapidly.

(5) Higher Quality: Agile development emphasizes a collaborative and iterative approach to quality assurance. Testing is integrated throughout the development process.

(3) What's are the 4-P's in project planning? Explain each of them briefly.

The 4-P's in project planning are.

People:- People are the primary key factor for successful organization. For the successful software production environment any or organization must perform the proper staffing, communication and coordination work environment, performance management, training, compensation, competency analysis and development, career development, workgroup development, team/culture development, and others. The people involve: The senior managers, Project managers, Analysts, Designers Software developers, testers and customers.

Product:- Product is any software that has to be developed. To develop successfully, product objectives and scope should be established; alternative solutions should be considered, and technical and management ~~or~~ constraints should be identified. Without this information, it is impossible to define reasonable and accurate estimate of the cost, an effective assement of risk, a realistic breakdown of project tasks or a manageable project schedule that provides a meaningful indication of progress.

③ Process: It is the set of framework activities and software engineering tasks to get the job done. The project manager is responsible for deciding what process to follow for doing the project and use appropriate SDLC model. A software process provides the framework from which a comprehensive plan for software development can established. A number of different tasks sets - tasks, milestones, work products, and quality assurance points - enable the framework activities to be adapted to the characteristics of the software project and the requirements of the project team. Finally, umbrella activities overlay the software process model.

④ Project: The project is the complete software project that includes requirement analysis, development, delivery, maintenance and updates. The project manager of a project or sub-project is responsible for managing the people, product and process. The responsibilities or activities of software project manager would be a long list but that has to be followed to avoid project failure.

Q) What is feasibility study in the context of software development? Why is it conducted?

Feasibility study refers to the process of evaluating the viability and potential success of a software project. It aims to determine whether the project is technically, economically, and operationally feasible before committing resources to its development.

There are different types of feasibility study.

① Technical Feasibility. This aspect assesses whether the required technology and infrastructure are available or can be developed to support the software project. Also, it must be feasible to afford the technical person for new techniques or features required on the ~~sft~~ software product.

② Schedule Feasibility: The software product must be built in time, lunched in market in proper time stamp for competition, and financially beneficial.

③ Financial Feasibility: The software product must be cost effective so that its user or users can purchase it.

④ Operational Feasibility: It assesses the extent to which the required software perform a series of steps to solve business problems & user requirements.

Determine whether the solution suggested by the software development team is acceptable.

⑤ Resource Feasibility:- The software development company must have the software and hardware resources to build the quality software product as the customer demand. Also the product must be well operating on the end user machine after delivery.

We need to do feasibility study for the following reason.

① Risk assesment:- A feasibility study helps identify potential risks and challenges associated with the software project.

② Resource allocation:- A feasibility study helps to allocate different types of resource such as Human, technical, Non-technical etc efficie

③ Decision making:- The feasibility study provides stakeholders with valuable information to make informed decisions.

④ Cost-Benefit analysis:- The economic feasibility analysis within the feasibility study helps asses the financial viability of the software project.

⑤ Stakeholder Alignment:- The feasibility study facilitates communication and alignment among project stakeholders.

(5) ~~Assurrest~~ How do you estimate software projects? Discuss the factors and techniques involved in software project estimation.

A general project estimation can be done by:

(1) Delaying estimation until late in the project.
(2) Estimation projects based on similar projects in the past.
(3) Using simple relatively decomposition technique.
(4) Using one or more empirical models for software cost and effort estimation.

First two techniques are not efficient.

(3) Decomposition technique:-
There are two approaches in decomposition technique
(i) Problem based estimation: Problem decomposed · LOC FP.
(ii) Process based estimation:- Process is decomposed into a relatively small set of tasks & efforts required to accomplish each task is estimated.

Factors involves in project estimation.

① Project Scope: The scope defines the deliverables, objectives, and boundaries of the project. A clear understanding of the project scope is crucial for accurate estimation.

② Requirements: The project requirements outline the features, functionalities and constraints of the project.

③ Project size: The size of the project refers to the amount of work that needs to be done. It can be measured in terms of lines of code, function points, user stories, or any other relavant metric.

④ Project complexity: Complex project with interdependencies, technical challenges, or novel ~~stat~~ solutions tends to require more time & effort to estimate and complete.

⑤ Team Experience & skill level: The experience & skill level of the project team members play a vital role in estimation.

Techniques involved in project estimation:

(1) Expert Judgement: This technique involves seeking input from subject matter experts or experienced professionals who have knowledge and expertise in similar projects.

(2) Analogous Estimating:- This techniques relies on historical data from similar projects as a basis for estimating the current project.

Date
Page

⑧ What are the key principles and practise of software engineering? How do they contribute to successful software development?

Software engineering is guided by a collection of core principles. SE principles help for software process and the executing software engineering methods effectively.
At the process level, core principles establish a philosophical fundation. It guides a software team as it perform framework and umbrella activities, navigates the process flow and produces a set of software engineering work products.
At the level of practice, core principles establish a collection of values and rules that serve as a guide as you analyze a problem, design a solution, implement and test the solution and ultimately deploy the software in the user community.

Basic principle ~~that~~ and practise of SE.

① Divide and conquer: A large problem is easier to solve if it is subdivided into a collection of elements (or concerns). Ideally, each concern delivers distinct functionality that can be developed, and in some cases validated, independently of other concerns.

(2) Understand the use of abstraction: An abstract is showing necessary details with hiding implementations. In analysis and design work, a software team normally begins with models that represent high level of abstraction and slowly refines those models into lower level of abstraction.

(3) Strive for consistency: Whether it's creating a requirements model, developing a software design, generating source code, or creating test cases, the principle of consistency suggests that a familiar context makes software easier to use.

(4) Focus on transfer of information: Software is about information transfer - from a database to an end user, legacy system to a WebApp, operating system to an application, etc. In every case of information flows, there are opportunities for error, or omission, or ambiguity.

(5) Build software that exhibits effective modularity: Any complex system can be divided into modules (components), each module should focus exclusively on one well-constrained aspect of the system - it should be cohesive in its function and or constrained in the content it represents.

6. Look for patterns:- The goal of patterns within the software community is to create a body of knowledge for future planning which defines our understanding of good architectures that meet the needs of their users.

7. When possible, represent the problem and its solution from a number of different perspectives. When a problem and its solution are examined from a number of different perspectives, it is more likely that greater insight will be achieved and that error and omissions will be uncovered.

8. Remember that someone will maintain the softw are! Over the long term, software will be corrected as defects are uncovered, adapted as its environment changes, and enhanced as stake holders request more capabilities. These maintenance activities can be facilitated if solid software engineering practice is applied throughout the software process.

Q7

Date
Page

Q7 Explain the concept of requirement engineering and its significance in software development.

The process of gathering the software requirements from the client, analyze and document is called Requirement engineering. The broad spectrum of tasks and techniques that lead to an understanding of requirements is called requirement engineering. The main goal of requirement engineering is to develop and maintain sophisticated and descriptive software Requirement Specification (SRS) document.

It consists of several process:

① Inception: A task that defines the scope and nature of the problem to be solved.

② Elicitation: A task that helps stakeholders define what is required. Problems of scope, problems of understanding, and problems of volatility and three problems that are encountered as elicitation occurs.

③ Elaboration: The information obtained from the customer during inception and elicitation is expanded and refined during elaboration. This task focuses on developing a refined requirements models that identifies various aspects of software function, behaviour, and information.

④ Negotation: This steps deals with the what are the higher priorities needs and what is essential and timestamp when it is needed.

⑤ Specification: The term specification means different things to different people. A specification can be a written document, a set of graphical models, a formal mathematical model, a collection of usage scenarios, a prototype, or any combination of these.

Significance of RE in Software Engineering.

① It helps to understands the actual needs of the user with collaboration & discussion.
② It helps in deping the scope of the software
③ Clear and well defined requirement helps to minimize project risks.
④ It helps in communication with & collaboration with project developers teams & stake holders.
⑤ It helps in change management during development.
⑦ It helps in Quality assurance & validation. Being well defined requirements it is easier to validate software if the software meets the specified criteria or not.

Date
Page

⑧ What is use case diagram & how is it used to model software requirements? provide an example.

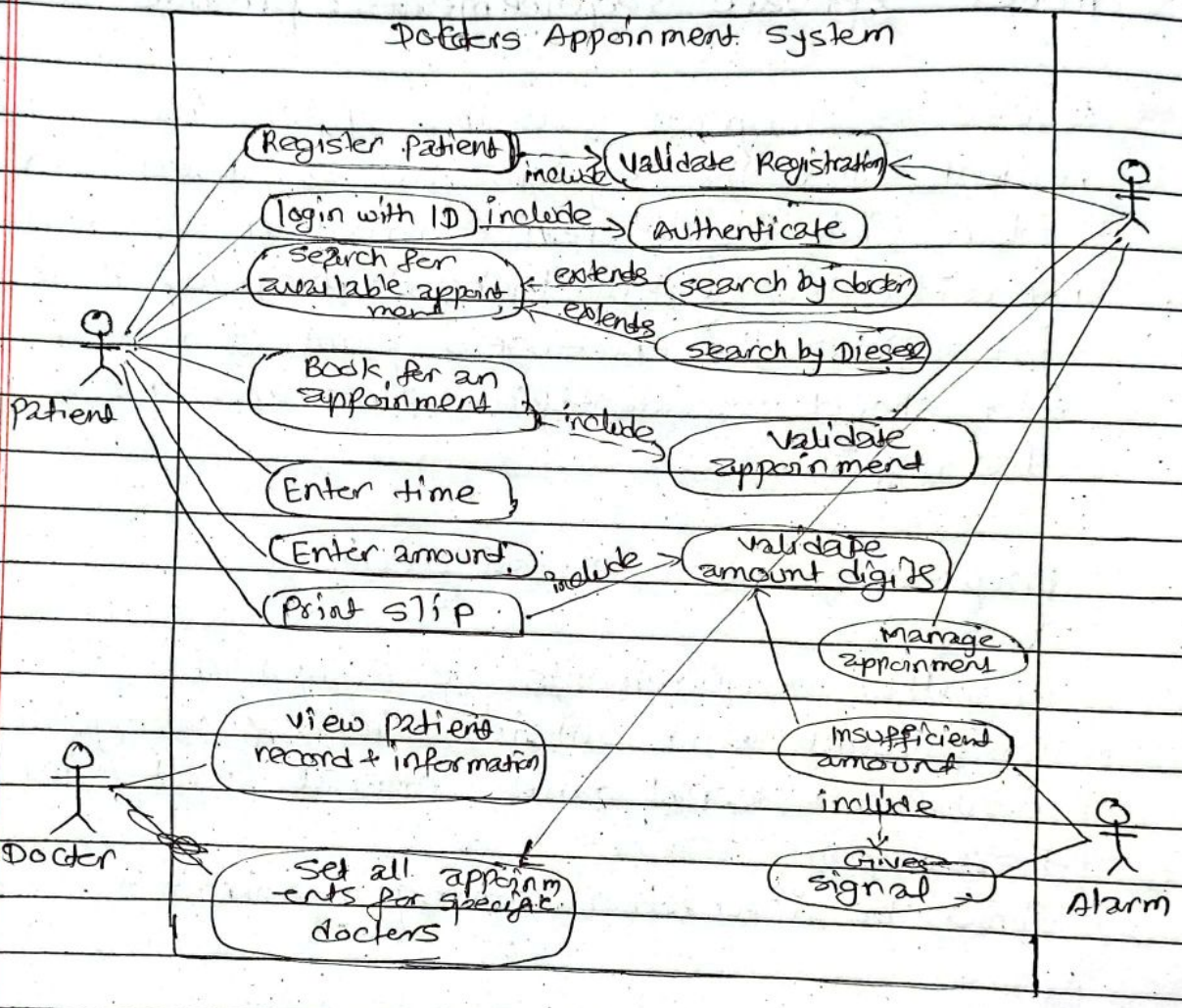
Use-case diagram are the system based description reflects how the system will be used and how the actors (i.e. People, device, program etc) will be dealing with the system to be developed. It models the system from the end-users point of views. So, the actor should be identified first and their roles are also defined.

Purpose of use-case diagrams:-

- To gather requirements of a system
- To present an outside view of a system
- Identify external and internal factor influencing the system.
- Show the interaction among requirements and actors.

Use case diagram actually defines ~~how~~ what system need to be develop and what must be the entity and how the entity interacts with the system and what process need to develops. So this provides a actual requirements of the system for the developers.

For example:-



Here from the above use-case diagram software gets the clear knoweledge about how many entities may be present in the system and how they interacts with the system and what functionalities are must be given to whom etc. So the use-case diagram provides the clear & visual idea about the requirement of the system.

Q) Discuss the purpose & benefits of ER diagrams, activity diagrams, class diagrams, and object diagrams in software design.

Purpose and Benefits of

(a) ER-diagram:

Purpose:-

→ Model the conceptual data scheme of the system.
→ Depict entities, attributes and relationships among entities in a database.

Benefits:-

→ Helps in understanding the structure and organization of data.
→ Assist in database design, Normalization, and data integrity.
→ Facilates effective communication among stakeholders.

(b) Activity Diagram:-

Purpose:-

→ Illustrate the flow of activities or process within a system.
→ Shows the sequence of activities, decisions and concurrency.

Benefits:
→ Visualize and understand complex scenarios and process flow.
→ Identify bottlenecks and optimize process efficiency.
→ Useful in business process modeling system, analysis and design.

(c) Class Diagrams:
Purpose:
→ Show instances of classes and their relationship at a specific time.

Benefits:
→ Understand and validate object behaviour & interaction.
→ Helps in testing, debugging and verifying system implementation.
→ Visualize communication & collaboration among developers.

(d) Object diagram.
Purpose
→ shows the instance of classes and their relationships at a specific point in time.
→ It provides a static view of the system, focusing on the objects and their interconnections.

→ It illustrate the structure & behaviour of a system or a specific scenario within the system.

Benefits

→ It shows how objects interact with each other and how they collaborate to acheive specific functionality.
→ By using object diagram, developers can identify design issues, such as incorrect relationships, missing or unnecessary objects, or potential bottlenecks in the system.
→ Object diagram can be used as a tool for testing and debugging.
→ Object diagrams serve as documentation artifacts that capture the structure and relationships of objects in a system.

(10) Explain the differences between DFD (Data Flow Diagram), Control Flow Diagrams, State diagrams, Sequence diagrams and collaboration diagrams. How are they used in the software design process.

(1) Data Flow Diagram (DFD):-

- Focus: Data flow and transformation within a system.
- Component: Processes, Data store and external entities.
- Purpose: Illustrate how data moves through a system.
- Usage: Requirements analysis system design and documentation.

(2) Control Flow Diagram (CFD):

- Focus: Data flow and sequence of action in a program.
- Components: statements, decision points, loops and branches.
- Purpose: Design and analyze program logic.
- Usage: Programming and software development

③ State Diagram:
- Focus: Objects or system behaviour and state transitions.
- Components: States, events, action and transitions.
- Purpose: Model the behaviour of objects or systems.
- Usages: Designing Complex systems, analyzing system behaviour and system developments.

④ Sequence Diagram:
- Focus: Interaction between object over time.
- Components: Objects, Lifelines, messages and time ordering.
- Purpose: Visualize dynamic behaviour and object interactions.
- Usage: System analysis, design & communication.

⑤ Collaboration Design:
- Fous: Relationships & interactions between objects.
- Components: Objects, messages, links and Associations.
- Purpose: illustrate object communication & relationships.
- Usage: System design, understanding object collaboration & communication.

→ DFDs are used in the early stages of Slw design to capture system requirements, understand the flow of data, and identify process interactio

→ CFDs are used in program design to plan and analyze the control flow logic, ensuring that the program executes as intended and to identify potential issues such as infinite loops or unreachable code.

→ State Diagrams are used to model and analyze the behaviour of complex systems, particularly those with a significant number of states and state transitions. They aid in designing and understanding the behaviour of objects or systems that have distinct states.

→ Sequence Diagrams are used to model and visualize the dynamic behaviour of a system. They capture interaction between objects and the order in which messages are exchanged, aiding in understanding the sequences of actions & collaboration between objects.

→ Collaboration diagram are used to illustrate the relationships & interactions between objects in a system communication.