SE

Assignment-I

1812401160 01
Vishwas Acharya

Page No. :
Date : / /

1) What is software, why we need software?

Ans. What is software?

↳ Software is a collection of instructions that enable the user to interact with a computer, its hardware, or perform tasks

↳ Without Software, most computers would be useless.

↳ E.g., without internet browses software, you could not surf the internet & without an operating system, the browsers could not run on your computer

- Why we Need Software?

↳ Software testing is really required to point out the defects and errors that were made during the development phases.

↳ It's essential since it makes sure of the customer's reliability and their satisfaction in the application

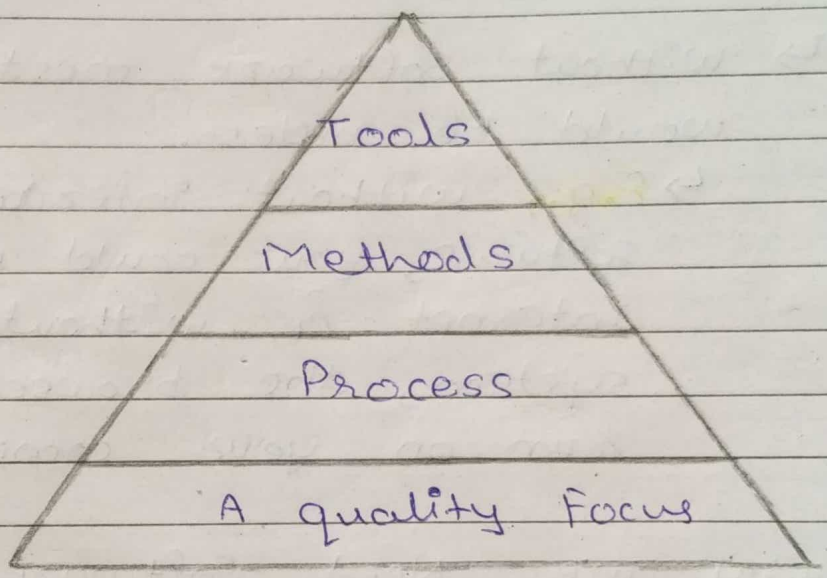
Page No. :
Date : / /

2) Explain Layered technology?

Ans • Software development is totally a layered technology. That means, to develop software one will have to go from one layer to another.

• The layers are related and each layer demands the fulfillment of the previous layer.

• Figure →



• A Quality Focus :-

- Every organization is rest on its commitment to quality.

- Total quality management, six segma or similar continuous improvement culture and it is this culture ultimately leads to development of increasingly more effective approches to software engineering.

- The foundation that supports software is a quality Focus.

- **Process :-**

- Process defines a framework that must be established for effective delivery of SE technology.

- The SE process is the glue that holds the technology layers together and enables rational and timely development of computer software

- **Methods :-**

- SE methods provide the technical aspects for building software

- SE methods rely on the set of modeling activities and other descriptive techniques.

- **Tools :-**

- SE tools provide automated or semiautomated support to the process & methods.

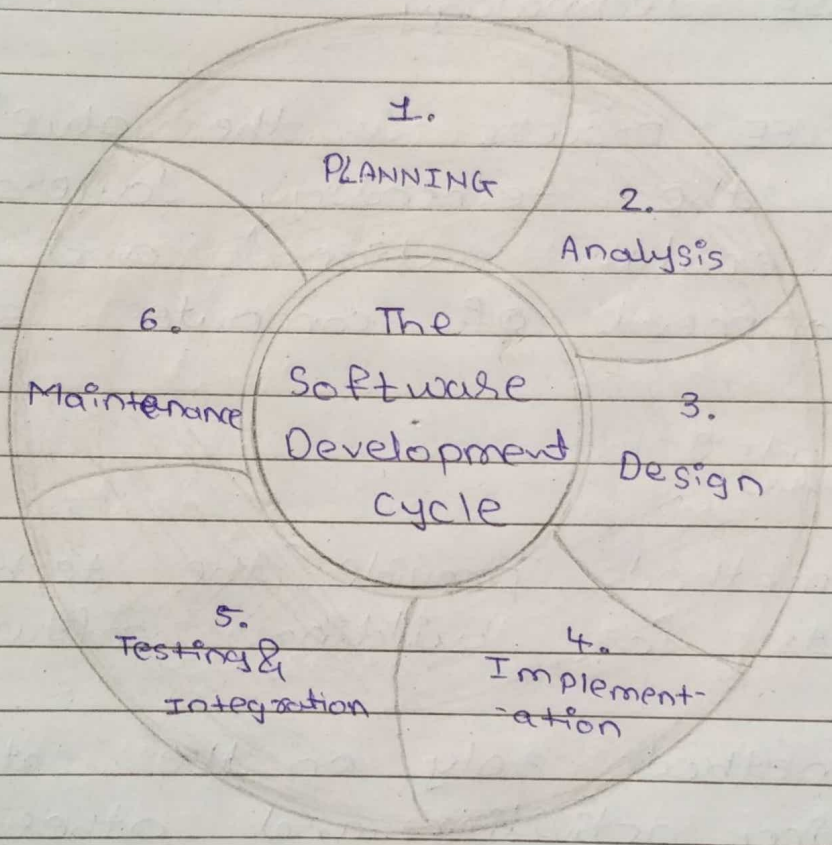
- When tools are integrated so that info created by one tool can be used by another a system for support of

Page No. :
Date : / /

software development called CASE (Computer - aided software Engineering)

3) Explain Software Development Life Cycle.

Ans There are six phases in every software development life cycle model:



1. Planning: without the perfect plan, calculating the strengths and weakness of the project, development of software is meaningless. Planning kicks off a project flawlessly and affects its progress positively.

2. Analysis: This step is about analyzing the performance of the software at various stages and making notes

an additional requirements. Analysis is very important to proceed further to the next step.

3. Design: Once the analysis is complete, the step of designing takes over, which is basically building the architecture of the project. This step helps remove possible flaws by setting a standard and attempting to stick to it.

4. Development & Implementation: The actual task of developing the software starts here with data recording going on in the background.
Once the software is developed, the stage of implementation comes in where the product goes through a pilot study to see if it's functioning properly.

5. Testing: The testing stage assesses the software for errors and documents bugs if there are any

6. Maintenance: Once the software passes through all the stages without any issues, it is to undergo a maintenance process wherein it

will be maintained and upgraded from time to time to adapt to changes.
Almost every software development Indian company follows all the six steps, leading to the reputation that the country enjoys in the software market today.

4) What are the characteristics of Good software.

Ans A software product can be judged by what it offers and how well it can be used.
This software must satisfy on the following grounds:

- Operational
- Transitional
- Maintenance

- Well-engineered and crafted software is expected to have the following characteristics:

- Operational

- This tells us how well software works in operations.
- It can be measured on:
  ↳ Budget

↳ Usability
↳ Efficiency
↳ Correctness
↳ Functionality
↳ Dependability
↳ Security
↳ Safety

- **Transitional**

- This aspect is important when the software is moved from one platform to another:

↳ Portability
↳ Interoperability
↳ Reusability
↳ Adaptability

- **Maintenance**

- This aspect briefs about how well a software has the capabilities to maintain itself in the ever-changing environment:

↳ Modularity
↳ Maintainability
↳ Flexibility
↳ Scalability

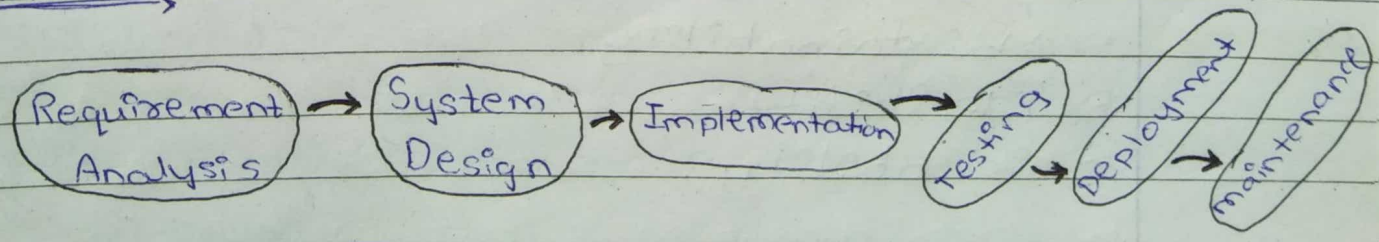
In Short, Software engineering is a branch of computer science, which uses well-defined engineering concepts required to produce efficient, durable, scalable, in-budget and on-time software products.

Sk Explain waterfall model, spiral model, prototype model.

Ans

Waterfall Model

- This model is the first process model to be introduced and is also known as Linear-sequential life cycle model.
- In this any phase in the development process begins only if the previous phase is complete, hence the phases do not overlap.
- This model is divided into separate phases and typically, the outcome of one phase acts as the input for the next phase sequentially.
- PHASES →



Page No. :
Date : / /

- Requirement Gathering & analysis : All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.

- System Design : Above phase is studied and then system design is prepared which helps in specifying hardware & ~~software~~ Sys. requirements & helps in overall sys. architecture.

- Implementation : with inputs from system design, the system is first developed in small programs called units, Each unit is developed & tested for its functionality, which is ~~re~~ referred to as Unit Testing.

- Integration & Testing : Here all the small units are integrated and then it is tested for any faults and failures.

- Deployment of System : After testing, the product is released into the market.

- Maintenance : Here those issues and better versions are handled & released in the client environment.

- **Advantages →**

↳ Phases are processed and completed one at a time.
↳ Process and results are well documented.
↳ Work well for smaller projects where requirements are very well understood.

- **Disadvantages →**

↳ Not a good model for complex and object-oriented projects.
↳ Poor model for long and ongoing projects.
↳ No working software is produced until late during the life cycle.

SPIRAL MODEL

- This model is an evolutionary software process model that couples that iterative nature of prototyping with controlled and systematic aspects of waterfall model.

- It provides potential for rapid develop^{t} of increasingly technology.

- It has four phases: Planning, Risk Analysis, Engineering and Evaluation.

- Figure →

- PLANNING : Requirements are gathered like BRS (Business Requirement Specification) & SRS (System Requirement Specification)

- RISK ANALYSIS : A process is undertaken to identify risk and alternate solutions. A prototype is produced at the end and if any risk is found the alternate solutions are suggested and implemented.

- ENGINEERING : Software is developed in this phase and testing at the end of the phase.

- Evaluation: This phase allows the customer to evaluate the output of the project to date before the project continues to the next spiral.

- Advantages

↳ Software is produced early in the software life cycle.
↳ Strong approval and documentation control.
↳ Additional functionality can be added at a later date.

- Disadvantages

↳ Can be costly model to use
↳ Doesn't work well for smaller project
↳ Project's success is highly dependent on the risk analysis phase

PROTYPE MODEL

- Before a design or coding proceed, a rough prototype is built to understand the requirements, and it is developed based on currently known requirements.

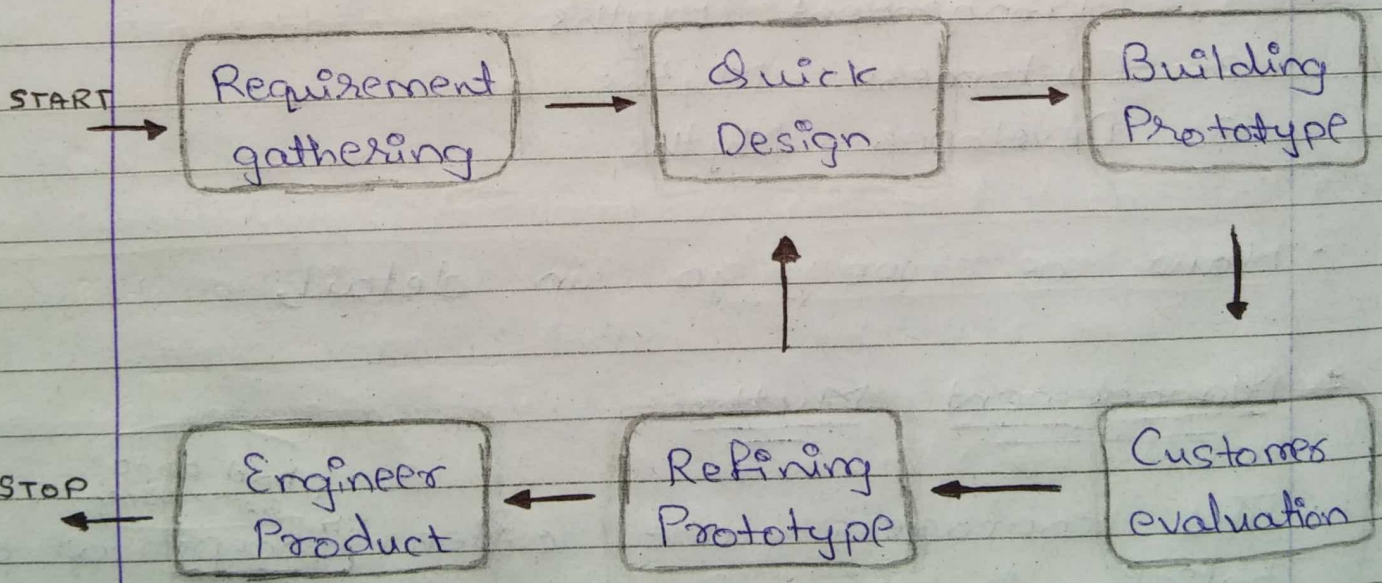
- It is a software model and using this client can get an "actual feel" of the system, and can get better

understanding of desired system.

- It is an attractive idea for complicated and large systems for which there is no manual process or existing system to help determining the requirements.

- Prototype usually not complete systems and many of the details are not built in the prototype. The goal is to provide a system with overall functionality.

- Figure →



- Advantages →

↳ Since a working model is provided, the users get a better understanding of the sys. being developed

↳ Errors can be detected much earlier
↳ Quicker user feedback is available leading to better solutions.

- **Disadvantages** →

↳ leads to implementing then repairing way of building systems.
↳ Incomplete application may cause application not to be ~~be~~ used as the full system was designed incomplete or inadequate problem analysis.

6) What are Software Myths?

Ans: These mainly three types of myths which are:

1. Management Myths
2. Customers Myths
3. Developers Myths

- Now as we go in detail,

1. **Management Myths**

- The managers are often (grasps) (hold on to something) at a belief in a software myth, same as a drowing person who grasps at a straw.

- Members acquires all the information:

- Generally, there is a myth that the members of the organization acquire all the information containing procedures, principles and standards.

- In reality, the developers don't have information about all the established standards because they are often outdated, incomplete and unadaptable. Plus, there is a rare chance that the developer will follow all the standards

- Adding more people can reduce time gap:

- It's another myth that more the number of programmers, lesser will be the time gap.

- If a project is behind the schedule, adding more manpower will further delay it because new workers will take more time to learn about the ongoing project.

- The management can relax themselves by outsourcing it's project:

- If an organization outsource its software to a third party, it does not relieve the management of its

duties. When the organization outsources the software project, they suffer invariably.

2. Customers Myths:

- The customers are encouraged by some marketing people in underestimating the difficulty of developing software.

- Software is malleable as a result of which changes are easy to accommodate:

- There is an enormous amount of labour required to have a change in the software after the release.

- It is not so easy to accommodate these changes.

- To start coding, a general statement of need is enough:

- The developers can't read the customer's mind and requires detailed descriptions of the requirements, in order to start coding.

3. Developer Myths:

- The software development art is becoming

an engineering discipline, but these are lots of myths.

- Once the code is delivered, the software can be called complete:

- In reality, more than 60% of the efforts are expended after the delivery of the software to the user.

- The software's success depends on the product's produced quality:

- The project does not become successful on the quality of the programs because both the software config. and documentation also play an important role in its success.

- The unnecessary documentation is required in software engineering, which further slows down the growth rate of a project

- The assessment of the software quality can be addressed after the execution of the program.

- The product, which is delivered after the project's completion can be called working programs.

Page No. :
Date : / /

7) What is Agile Principles?

Ans Following are the agile manifesto principles:

(1) Individuals and Interaction

↳ In agile development, self-organization and motivation are important, as are interactions like co-location and pair programming.

(2) Working Software

↳ Demo working software is considered the best means of communication with the customers to understand their requirements, instead of just depending on documentation.

(3) Customer collaboration

↳ As the requirements cannot be gathered completely in the beginning of the project due to various factors, continuous customer interaction is very important to get proper product requirements.

(4) Responding to change

↳ Agile Development is focused on quick responses to change and continuous development.

Page No. :
Date : / /

Q. What is Scrum? Describe its development activities?

Ans
- Scrum principles are consist of agile manifesto and are used to guide development activities with a process that incorporates the five framework activities.

- Within each framework activity, work tasks occur within a process pattern called a sprint.

- Scrum emphasizes the use of set of software process pattern that have proven effective projects with tight timelines.

- Items can be added to backlog at any time.

- Product manager assesses the backlog and updates priorities as required.

- Advantages →

- large projects are broken down into small sprints.

- customers are involved, so best result are ensured.

- Budget friendly and time saving
- Developments are coded and tested during the sprint review.

- **Disadvantages ⇒**

- Once decided the sprint backlog items cannot be changed
- Requires strong commitment from all members of the team.
- Quality is hard to impliment until team goes through aggressive testing process.

Q) Why do we require Requirement Gathering for development of any project?

Ans. Requirement gathering is a fundamental part of any software development project.

- It is foundation on which all projects should built.
- The gathering and compiling of requirements for a software project is very much a partnership between the users of the S/W & the developers.

Page No. :
Date : / /

- The importance of requirements gathering is to business customers tend to expect software teams to deliver a solution based on unspoken, incomplete or unknown requirements, while software teams tend to assume that business customers will communicate exactly what they want as succinctly as possible.

- Both expectation are obviously unrealistic.

- Therefore, the requirements need to be formally captured in one document that can be used as a reference during software development.

- Good gathering, processing & management of requirement is important as it sets ~~de~~ clear targets for everyone to aim for.

- The requirements gathering should detail how a user would accomplish what they want using the software being developed.

- Thus, we require requirement gathering for developement of any project.

Page No. :
Date : / /

10) Is planning is Important ? If yes then state it's advantage ?

Ans. Yes, planning is important.

- Before starting software project, it is essential to determine the tasks to be performed & properly manage allocation of tasks among individuals involved in the software development.

- It also determines project constraints, roles & responsibilites of the project, checks feasibility of the schedule & uses requirements.

- Hence, planning is important as it results in effective software developmed.

◉ Advantages of Planning :-

↳ Focuses attention on objectives & results.
↳ Attention on objectives
↳ establishes a basis for teamwork
↳ Minimizing uncertainties
↳ Better utilization of resources
↳ Economy in operations
↳ Better co-ordination
↳ Encourages Innovations & Creativity
↳ Facilitates control, delegation
↳ Management by exception possible