ASDM

ASSIGNMENT NO -01

- Name: Nilesh Patil
- Batch: A

Q1. Discuss the prototyping model. What is the effect of designing a prototype on the overall cost of the project?

Ans: Prototyping may have some initial costs of developing, but it reduces the overall budget by helping your product to be free of the errors or glitches that could have occurred if the idea was made from scratch without any prior user testing. If the prototyping process is ignored completely, it might result in the restructuring and redesigning of the entire product after spending all your resourceson its development. So, the effect of designing a prototype on theoverall cost of a software project is to actually reduce the additional costs of restructuring and reframing it after its full-fledged development- which might cost a fortune

- Requirements gathering: The requirements for the software are identified and documented.
- Prototype design: A prototype of the software is designed based on the identified requirements.
- Prototype development: The prototype is developed using rapid application development (RAD) techniques.
- Prototype testing: The prototype is tested and evaluated to identify any design flaws or issues.
- Prototype refinement: Based on the feedback received from testing, the prototype is refined and improved.
- Final product development: The final product is developed based on the refined prototype design.

Q2. Compare iterative enhancement model and evolutionary process model.

Ans:

The Iterative Enhancement Model and the Evolutionary Process Model are both software development methodologies that focus on iterative development and continuous improvement. However, there are some key differences between these two models:

1. Scope: The Iterative Enhancement Model focuses on delivering a fully functional system in small increments. Each increment is developed through a sequence of design, implementation, testing, and evaluation phases. The Evolutionary Process Model, on the other hand, focuses on developing a prototype that can be improved through successive iterations.

2. Requirements: In the Iterative Enhancement Model, the requirements are usually well defined and stable. The Evolutionary Process Model, on the other hand, assumes that the requirements will evolve over time as the prototype is developed and refined.

3. Feedback: The Iterative Enhancement Model relies on feedback from end-users to refine and improve each increment. The Evolutionary Process Model relies on feedback from end-users to refine and improve the prototype.

4. Flexibility: The Iterative Enhancement Model is more rigid and structured, with a set process and sequence of phases for each increment. The Evolutionary Process Model is more flexible and adaptive, allowing for changes in the requirements and design as the prototype evolves.

5. Risks: The Iterative Enhancement Model is less risky since each increment is fully functional and tested before moving on to the next increment. The Evolutionary Process Model is more risky since the prototype is often incomplete and may require significant changes based on feedback from end-users.

Q3. As we move outward along with process flow path of the spiral model, what can we say about software that is being developed or maintained.

Ans: The Spiral Model is a software development process model that is characterized by iterative cycles of planning, risk analysis, prototyping, and testing. The process flow path of the Spiral Model is represented by a spiral, where each loop of the spiral represents a new iteration of the software development process. As we move outward along the process flow path of the Spiral Model, we can say that the software being developed or maintained is becoming more refined and mature. Each iteration of the spiral adds new functionality, refines existing functionality, and improves the quality of the software.

In the earlier iterations of the Spiral Model, the focus is on planning and risk analysis. The software being developed or maintained is in the initial stages of development, and the primary goal is to identify potential risks and develop strategies to mitigate them.

As we move further outward along the process flow path of the Spiral Model, the focus shifts to prototyping and testing. The software being developed or maintained is becoming more functional and closer to the final product. Prototyping and testing allow for feedback from end-users and stakeholders, which can be used to refine and improve the software.

In the later iterations of the Spiral Model, the focus is on finalizing the software and preparing it for deployment. The software being developed or maintained is in the final stages of development, and the primary goal is to ensure that it meets all requirements and is of high quality.

Overall, as we move outward along the process flow path of the Spiral Model, we can say that the software being developed or maintained is becoming more refined, functional, and closer to the final product. Each iteration of the Spiral Model adds value to the software and helps to ensure its success.

Q4. Explain the Scrum Agile methodology.

Ans:

Scrum is an Agile methodology for software development that emphasizes teamwork, collaboration, and flexibility. It is based on an iterative and incremental approach to software development and is designed to deliver working software in a timely and efficient manner. The Scrum Agile methodology is based on three main roles: the Product Owner, the Scrum Master, and the Development Team.

The Product Owner is responsible for defining and prioritizing the features and requirements of the software being developed. The Product Owner works closely with the Development Team to ensure that the software meets the needs of the end-users.The Development Team is responsible for designing, building, and testing the software. The Development Team is self-organizing and cross-functional, meaning that they work together to complete all aspects of the software development process.

The Scrum Agile methodology is based on a series of time-boxed iterations called Sprints. Each Sprint typically lasts two to four weeks and includes the following phases:

1. Sprint Planning: The Product Owner and the Development Team work together to define the goals and requirements for the Sprint.
2. Daily Scrum: The Development Team meets daily to discuss their progress and identify any obstacles that may be preventing them from achieving their goals.
3. Sprint Review: The Development Team presents the completed work to the Product Owner and other stakeholders, who provide feedback and suggest changes.
4. Sprint Retrospective: The Development Team reflects on the Sprint and identifies ways to improve their processes and work more effectively.

The Scrum Agile methodology emphasizes communication, collaboration, and transparency. It is designed to be flexible and adaptive, allowing for changes in requirements and priorities as the project progresses. By focusing on delivering working software in short iterations, Scrum helps to reduce risk, increase efficiency, and improve the quality of the software being developed.

Q5. Explain the utility of Kanban CFD reports.

Ans:

Kanban is an Agile methodology that emphasizes visualizing work, limiting work in progress, and maximizing flow. A key tool used in Kanban is the Cumulative Flow Diagram (CFD) report, which provides valuable insights into the flow of work and helps teams to identify potential bottlenecks and areas for improvement.

The CFD report tracks the flow of work through various stages of the development process, from backlog to done. It shows the number of items in each stage at a given point in time and the rate at which items are moving between stages. The CFD report can be used to answer questions such as:

- How much work is in progress?
- How quickly are items moving through each stage of the process?
- Are there any bottlenecks or areas of congestion?
- Are we meeting our goals for work completion?

The CFD report is a useful tool for Kanban teams because it provides a visual representation of the flow of work and helps to identify areas for improvement. By analyzing the CFD report, teams can identify bottlenecks and areas of congestion, and take steps to improve the flow of work. For example, if the CFD report shows that there are too many items in progress, the team may decide to limit work in progress to improve flow. Similarly, , the team may investigate the root cause and take steps to address it.

Overall, the CFD report is a valuable tool for Kanban teams because it provides a clear and concise view of the flow of work, and helps teams to identify potential problems and areas for improvement. By using the CFD report to track progress and make data-driven decisions, Kanban teams can improve their processes and deliver high-quality work more efficiently.