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

▪ Prototyping basically means to practically implement the findings of a discovery phase in design development. It means to develop a model or a dummy website for a product that is partially functional and gives a great tangible idea of how the product will look and feel after its development. Prototyping helps a user to–

▪ Shaping your ideas– Prototyping helps in improving the creative process by starting with the idea and iterating that idea over and over until it satisfies your needs.

▪ Communicate your ideas for meeting the client's business objectives– It helps to communicate the idea if it does not exactly meet the client's business objectives, and you can avoid the complete restructuring of the developed product by just revising the product prototype.

▪ Coming up with solutions quickly– You and your team can easily collaborate and use the cloud space for designing the prototype together and leave comments and suggestions for fixing the loopholes in the prototype.

▪ Building a bridge between the designers and customers– Prototypes help the designers to understand the customer's tastes and preferences, according to which they slowly realize which product can lure which customer, and improve communication between them.

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. Furthermore, prototyping also helps to understand the intrinsic flaws, shortcomings and drawbacks that can be improved during the product development process. If the prototyping process is ignored completely, it might result in the restructuring and redesigning of the entire product after spending all your resources on its development. So, the effect of designing a prototype on the overall cost of a software project is to actually reduce the additional costs of restructuring and re-framing it after its full-fledged development- which might cost a fortune.

Q2) Compare iterative enhancement model and evolutionary process model.

Evolutionary Process Model

Evolutionary process model resembles the iterative enhancement model. The same phases are defined for the waterfall model occurs here in a cyclical fashion. This model differs from the iterative enhancement model in the sense that this does not require a useful product at the end of each cycle. In evolutionary development, requirements are implemented by category rather than by priority.

For example, in a simple database application, one cycle might implement the graphical user Interface (GUI), another file manipulation, another queries and another updates. All four cycles must complete before there is a working product available. GUI allows the users to interact with the system, file manipulation allow the data to be saved and retrieved, queries allow user to get out of the system, and updates allows users to put data into the system.

Benefits of Evolutionary Process Model

- Use of EVO brings a significant reduction in risk for software projects.
- EVO can reduce costs by providing a structured, disciplined avenue for experimentation.
- EVO allows the marketing department access to early deliveries, facilitating the development of documentation and demonstration.
- Better fit the product to user needs and market requirements.
- Manage project risk with the definition of early cycle content.
- Uncover key issues early and focus attention appropriately.
- Increase the opportunity to hit market windows.
- Accelerate sales cycles with early customer exposure.
- Increase management visibility of project progress.
- Increase product team productivity and motivations.

Iterative Development Model

Iterative development is an approach to building software (or anything) in which the overall life cycle is composed of several iterations in sequence. Each iteration is a self-contained mini-project composed of activities such as requirements analysis, design, programming, and test. The goal for the end of an iteration is an iteration release, a stable, integrated and tested partially complete system. To be clear all the software across all the teams is integrated into a release each iteration. Most iteration releases are internal, a baseline primarily for the benefit of the development team they are not released externally. The final iteration release is the complete product, released to the market or clients.

Benefits of Iterative Enhancement Model

- A limited number of persons can be put on the project because work is to be delivered u parts.
- This model can result in better testing as testing each increment is likely to be easier than testing the whole developed software system.
- This process has the major advantage that the customer does not have to pay for the entire software at a time.
- The project is to be delivered in parts; the total cost of the project is distributed.
- Customers or end-users get the choice to see the useful functionality only in the software development life cycle.
- End user's feedback requirements for successive releases become clearer.
- As usefulness is augmented in advances, testing additionally turns out to be simple.
- The hazard of disappointment with the product is diminished as clients begin utilizing the item early.

Q.3 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.

Spiral Model in Software Development Life Cycle

- The First step is to start with an idea. This is usually the phase when developers or entrepreneurs come up with new ideas so that they can create a new product or software. A team of skilled personals is also selected for executing this project effectively and efficiently.
- The second step is to do a feasibility study on the idea. The feasibility study will help you in determining the cost of software development required to build it, the time is taken for developing it, and its functionality as well.
- The Third step is to create a business plan based on feasibility study results. The business plan should clearly define the Target Audience, Market Share, Estimated Cost of Software Development, etc.
- The Fourth step is to develop the initial prototype version for the product which will help you in proving the feasibility of your idea.
- Fifth step is to do User Acceptance Testing (UAT) on the prototype version. This testing should be performed by target users as well. If feedback from UAT is positive then the next step is identifying and resolving risks which we already discussed here.
- Once you are done with resolving risks then you can proceed to the final version of the software. This is the phase when you will work on adding more features as per your business plan or user requirements.
- Once this process is done then final testing should be performed and all defects if any should be fixed.
- Finally, the product will have to be deployed so that users can use it.
- User feedback on the final product should be collected and analyzed in order to identify any further enhancement required in the future. This is exactly where the spiral model comes into the picture, as you will go back to the first phase of this model whenever your customer wants to change something or add new features later on.
- The cycle can be closed when all user requirements are satisfied or the business model comes to an end.

Looking at the steps involved in using the spiral model in SDLC, it is very clear that there will be a lot of hurdles until you reach your objective. But this is where the true value of this model lies, as it appropriately addresses these hurdles in the right manner and makes the process of software development extremely efficient.

Q4. Explain the Scrum Agile methodology.

Scrum is an agile development methodology used in the development of Software based on an iterative and incremental processes. Scrum is adaptable, fast, flexible and effective agile framework that is designed to deliver value to the customer throughout the development of the project. The primary objective of Scrum is to satisfy the customer's need through an environment of transparency in communication, collective responsibility and continuous progress. The development starts from a general idea of what needs to be built, elaborating a list of characteristics ordered by priority (product backlog) that the owner of the product wants to obtain.

Planning in Scrum

The Sprint Planning Meeting is held at the beginning of each Sprint. All the members of the Team participate in the meeting, i.e., the Product Owner, Scrum Master and all the Development Team. The entire Scrum team must understand and define what objective should be obtained in that Sprint (Sprint Goal). From this point the development team must design a work plan to achieve the objective. This planning should allow you to see if the sprint goal involves a workload according to the duration stipulated for the Sprints (which is 2 to 4 weeks).

The client shows the result to be achieved in that Sprint and the requirements of the deliverable product. Here you have to carry out a discussion in which the development team evaluates what elements of the list can be delivered.

Events in Scrum

Each of the Scrum events facilitates the adaptation of some of the aspects of the process, the product, progress or relationships.

Sprint:

Sprint is the basic unit of work for a Scrum team. This is the main feature that marks the difference between Scrum and other models for agile development.

Sprint Planning:

The goal of the Sprint Planning is to define what is going to be done in the Sprint and how it is going to be done. This meeting is held at the beginning of each Sprint and is defined how it will approach the project coming from the Product Backlog stages and deadlines. Each Sprint is composed of different features.

Daily Scrum:

The objective of the Daily Scrum is to evaluate the progress and trend until the end of the Sprint, synchronizing the activities and creating a plan for the next 24 hours. It is a brief meeting that takes place daily during the Sprint period. Three questions are answered individually: What did I do yesterday? What am I going to do today? What help do I need? The Scrum Master should try to solve problems or obstacles that arise.

Sprint Review:

The goal of the sprint review is to show what work has been completed with regards to the product backlog for future deliveries. The finished sprint is reviewed, and there should already be a clear and tangible advancement in the product to present to the client.

Sprint Retrospective:

The team reviews the completed goals of the finished sprint, write down the good and the bad, so as not to repeat the mistakes again. This stage serves to implement improvements from the point of view of the development process. The goal of the sprint retrospective is to identify possible process improvements and generate a plan to implement them in the next Sprint.

These are some of the collective benefits of agile scrum methodology:

- Creativity and innovation
- Lower costs • Quality improvement
- Organizational synergy
- Employee satisfaction
- Customer satisfaction
- Flexibility and adaptability

Agile Scrum Methodology Has Several Befits

First, it encourages products to be built faster, since each set of goals must be completed within each sprint's time frame. It also requires frequent planning and goal setting, which helps the scrum team focus on the current sprint's objectives and increase productivity. Agile is a process that allows a team to more efficiently manage a project by breaking it down into several stages, each of which allows for consistent collaboration with stakeholders to promote steady improvements at every stage. Agile was first described in the Agile Manifesto in 2000 by a group of developers who sought out a new method of writing software.

The manifesto Cites Four Values

1. Individuals and interactions over processes and tools .
2. Working software over comprehensive documentation .
3. Customer collaboration over contract negotiation .
4. Responding to change over following a plan .

In short, scrum is a framework for effective collaborations among teams working on complex products. Scrum is a type of agile technology that consists of meetings, roles, and tools to help teams working on complex projects collaborate and better structure and manage their workload. Although it is most often used by software development teams, scrum can be beneficial to any team working toward a common goal.

5. Explain the utility of Kanban CFD reports.

The cumulative flow diagram (also known as CFD) is one of the most advanced Kanban and Agile analytics charts. It provides a concise visualization of the three most important metrics of your flow:

- Cycle time
- Throughput
- Work in progress

Its main purpose is to show you how stable your flow is and help you understand where you need to focus on making your process more predictable. It gives you quantitative and qualitative insight into past and existing problems and can visualize massive amounts of data.

What is Kanban Cumulative Flow Diagram

The cumulative flow diagram tracks the total number of work items in progress each day. It is called "cumulative" because the values are accumulated over time.

The cumulative flow diagram shows tasks distribution along the process stages. The graph is built from colored bands of tasks, with each band indicating how many tasks are present in each stage of the process at a given time.

The vertical axis shows the number of tasks, while time is plotted on the horizontal axis. The curves are essentially the number of items in any process state, shown over time.

From the key, we have 3 groups of tasks:

- To Do
- In Progress
- Done

These terms are the basic process steps that are the core of any Kanban board. Whenever your team pulls a new task and starts working on it, it is applied to the CFD by incrementing the “In Progress” curve by one. Logically, the “Done” curve is only increased once the task has been completed and delivered.

The Kanban cumulative flow diagram is built up slowly – by tracking and accumulating every task which enters your project workflow, the diagram shows the clear flow of all tasks through your process. When a problem occurs, rather than staying smooth and gently rising, the graph will show a sudden jump upwards. The transition can be spotted instantly, making the CFD a powerful tool for every Kanban team.

Reading Cumulative Flow Diagram

There is a lot of critical flow information that can be drawn from the CFD at a glance. You can analyze your process in more depth by learning to recognize the most common CFD patterns. By tracking the duration of each task, teams can see how quickly they are delivering work.

The gradient of the CFD curves is used to observe changes in work in progress amounts. A sharp increase in the slope gradient would occur if your team is doing too many things at once. Such types of impediments visually show how stable your process is.

Work in Progress

One cornerstone of the Kanban method is that teams limit Work in Progress (WIP) to improve the quality and speed of delivery. The larger your WIP, the more tasks your team is attempting to complete simultaneously. This usually leads to constant context switching – a major source of low productivity and progress stagnation.

If the “work-in-progress” area is constant or decreasing with time, this is usually a positive sign for your team: you are meeting demand and delivering your product at a consistent pace. If WIP is growing, this may suggest a bottleneck or some other problem stemming from the workflow.

When examining any Kanban CFD, the first area to check up on is always work in progress. Remember to account for any changes in work conditions (team

size, change of personnel) before leaping to conclusions – these could provide explanations for unexpected spikes in activity.

Average Throughout

When looking at a CFD, you can check how much is actually being achieved. In other words, you can evaluate how steep the "Done" area is. If the team is very productive and well-suited to the job, they will be able to complete tasks quickly: the "Done" pile will quickly grow.

You can easily calculate your exact average throughput for a certain time range. If the bottom line of your CFD represents the "Done" state from your process, then the slope of that line between any two points is your average throughput between those two points.

Approximate Average Cycle Time

On the other hand, if we fail to do a good job, then the "Done" area would be flatter. If we are slow to deliver, then it means that our cycle times are larger. Reducing the cycle time, without sacrificing the quality of output, is a constant goal for Kanban teams. The faster a product is delivered, the more satisfied the client is.

You can measure approximate average cycle time by calculating the horizontal difference between the top line and bottom line of a CFD at any point along the graph. The cycle time is a strong indicator of the efficiency of work processes since it reflects how quickly a project is completed after work has begun. This is directly related to the work in progress, and shortening cycle times usually means increased customer satisfaction.