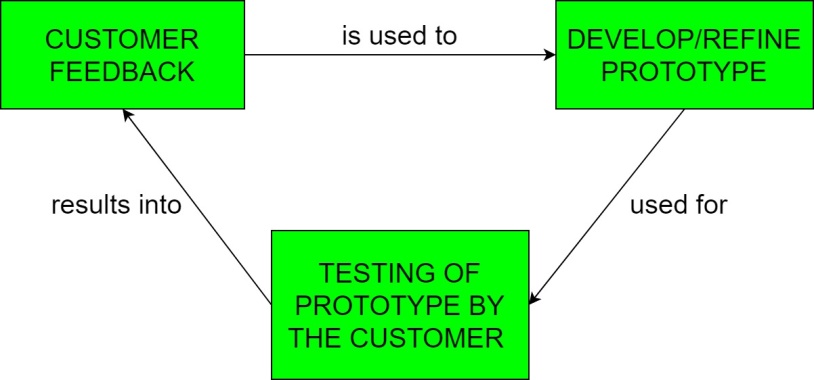
ASSIGNMENT ON SDLC MODEL

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

Prototyping is defined as the process of developing a working replication of a product or system that has to be engineered. It offers a small scale facsimile of the end product and is used for obtaining customer feedback as described below:



The Prototyping Model is one of the most popularly used Software Development Life Cycle Models (SDLC models). This model is used when the customers do not know the exact project requirements beforehand. In this model, a prototype of the end product is first developed, tested and refined as per customer feedback repeatedly till a final acceptable prototype is achieved which forms the basis for developing the final product.

In this process model, the system is partially implemented before or during the analysis phase thereby giving the customers an opportunity to see the product early in the life cycle. The process starts by interviewing the customers and developing the incomplete high-level paper model. This document is used to build the initial prototype supporting only the basic functionality as desired by the customer. Once the customer figures out the problems, the prototype is further refined to eliminate them. The process continues until the user approves the prototype and finds the working model to be satisfactory. There are four types of models available:

**a. Rapid Throwaway Prototyping**

**b. Evolutionary Prototyping**

**c. Incremental Prototyping**

**d. Extreme Prototyping**

**Advantages –** 

* The customers get to see the partial product early in the life cycle. This ensures a greater level of customer satisfaction and comfort.
* New requirements can be easily accommodated as there is scope for refinement.
* Missing functionalities can be easily figured out.
* Errors can be detected much earlier thereby saving a lot of effort and cost, besides enhancing the quality of the software.
* The developed prototype can be reused by the developer for more complicated projects in the future.
* Flexibility in design.

**Disadvantages –** 

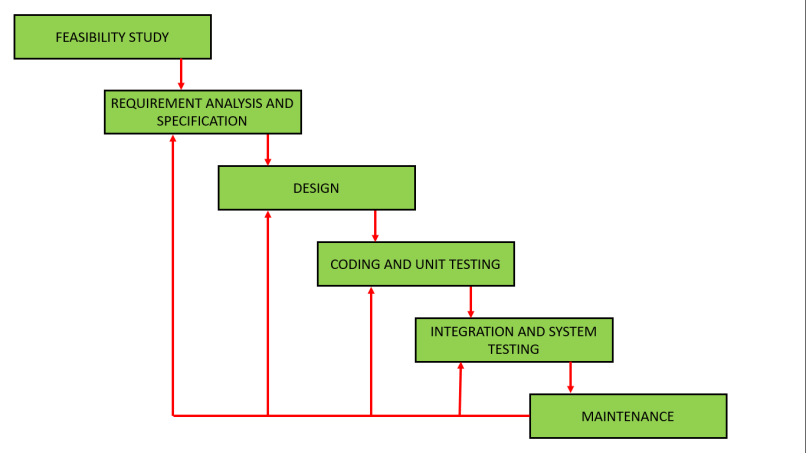
* Costly w.r.t time as well as money.
* There may be too much variation in requirements each time the prototype is evaluated by the customer.
* Poor Documentation due to continuously changing customer requirements.
* It is very difficult for developers to accommodate all the changes demanded by the customer.
* There is uncertainty in determining the number of iterations that would be required before the prototype is finally accepted by the customer.
* After seeing an early prototype, the customers sometimes demand the actual product to be delivered soon.
* Developers in a hurry to build prototypes may end up with sub-optimal solutions.
* The customer might lose interest in the product if he/she is not satisfied with the initial prototype.

Q2. Compare iterative enhancement model and evolutionary process model.

The Iterative waterfall model can be thought of as incorporating the necessary changes to the classical waterfall model to make it usable in practical software development projects. It is almost the same as the classical waterfall model except some changes are made to increase the efficiency of the software development.

The iterative waterfall model provides feedback paths from every phase to its preceding phases, which is the main difference from the classical waterfall model.

Feedback paths introduced by the iterative waterfall model are shown in the figure below.



When errors are detected at some later phase, these feedback paths allow correcting errors committed by programmers during some phase. The feedback paths allow the phase to be reworked in which errors are committed and these changes are reflected in the later phases. But, there is no feedback path to the stage – feasibility study, because once a project has been taken, does not give up the project easily.

It is good to detect errors in the same phase in which they are committed. It reduces the effort and time required to correct the errors.

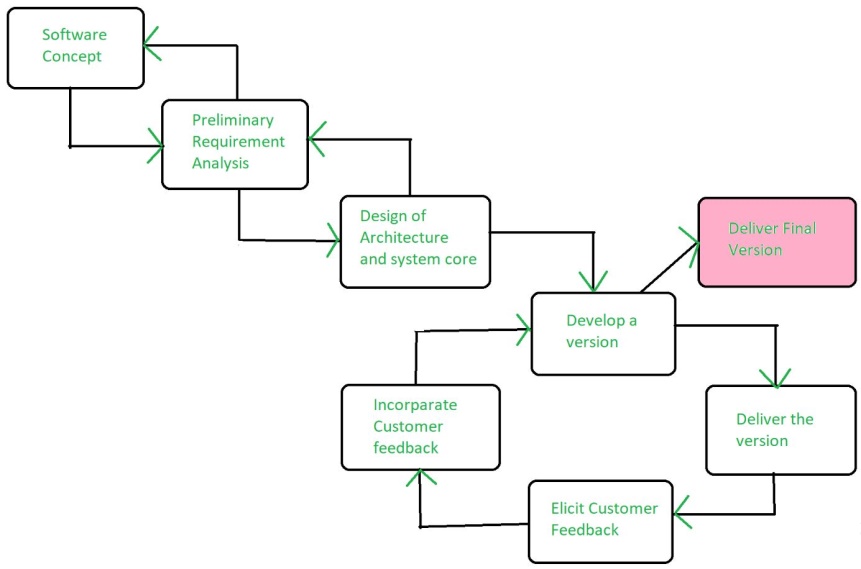
**Advantages of Iterative Waterfall Model :**

**a. Feedback Path**

**b. Simple**

**c. Cost-Effective**

**d. Well-organized**

**Evolutionary model** is a combination of [Iterative](https://www.geeksforgeeks.org/software-engineering-iterative-waterfall-model/)and [Incremental model](https://www.geeksforgeeks.org/software-engineering-incremental-process-model/) of software development life cycle. Delivering your system in a big bang release, delivering it in incremental process over time is the action done in this model. Some initial requirements and architecture envisioning need to be done. It is better for software products that have their feature sets redefined during development because of user feedback and other factors. The Evolutionary development model divides the development cycle into smaller, incremental waterfall models in which users are able to get access to the product at the end of each cycle. Feedback is provided by the users on the product for the planning stage of the next cycle and the development team responds, often by changing the product, plan or process. Therefore, the software product evolves with time. All the models have the disadvantage that the duration of time from start of the project to the delivery time of a solution is very high. Evolutionary model solves this problem in a different approach.   
   
Evolutionary model suggests breaking down of work into smaller chunks, prioritizing them and then delivering those chunks to the customer one by one. The number of chunks is huge and is the number of deliveries made to the customer. The main advantage is that the customer’s confidence increases as he constantly gets quantifiable goods or services from the beginning of the project to verify and validate his requirements. The model allows for changing requirements as well as all work in broken down into maintainable work chunks.

**Application of Evolutionary Model:**

1. It is used in large projects where you can easily find modules for incremental implementation. Evolutionary model is commonly used when the customer wants to start using the core features instead of waiting for the full software.
2. Evolutionary model is also used in object oriented software development because the system can be easily portioned into units in terms of objects.

**Necessary conditions for implementing this model:-**

* Customer needs are clear and been explained in deep to the developer team.
* There might be small changes required in separate parts but not a major change.
* As it requires time, so there must be some time left for the market constraints.
* Risk is high and continuous targets to achieve and report to customer repeatedly.
* It is used when working on a technology is new and requires time to learn.

**Advantages:**

* In evolutionary model, a user gets a chance to experiment partially developed system.
* It reduces the error because the core modules get tested thoroughly.

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.

**Spiral model** is one of the most important Software Development Life Cycle models, which provides support for **Risk Handling**. In its diagrammatic representation, it looks like a spiral with many loops. The exact number of loops of the spiral is unknown and can vary from project to project. Each loop of the spiral is called a **Phase of the software development process.** The exact number of phases needed to develop the product can be varied by the project manager depending upon the project risks. As the project manager dynamically determines the number of phases, so the project manager has an important role to develop a product using the spiral model.

The Radius of the spiral at any point represents the expenses(cost) of the project so far, and the angular dimension represents the progress made so far in the current phase.

**The below diagram shows the different phases of the Spiral Model: –**



Each phase of the Spiral Model is divided into four quadrants as shown in the above figure. The functions of these four quadrants are discussed below-

1. **Objectives determination and identify alternative solutions:** Requirements are gathered from the customers and the objectives are identified, elaborated, and analyzed at the start of every phase. Then alternative solutions possible for the phase are proposed in this quadrant.
2. **Identify and resolve Risks:** During the second quadrant, all the possible solutions are evaluated to select the best possible solution. Then the risks associated with that solution are identified and the risks are resolved using the best possible strategy. At the end of this quadrant, the Prototype is built for the best possible solution.
3. **Develop next version of the Product:** During the third quadrant, the identified features are developed and verified through testing. At the end of the third quadrant, the next version of the software is available.
4. **Review and plan for the next Phase:** In the fourth quadrant, the Customers evaluate the so far developed version of the software. In the end, planning for the next phase is started.

**Risk Handling in Spiral Model**  
A risk is any adverse situation that might affect the successful completion of a software project. The most important feature of the spiral model is handling these unknown risks after the project has started. Such risk resolutions are easier done by developing a prototype. The spiral model supports coping up with risks by providing the scope to build a prototype at every phase of the software development.

ThePrototyping model also supports risk handling, but the risks must be identified completely before the start of the development work of the project. But in real life project risk may occur after the development work starts, in that case, we cannot use the Prototyping Model. In each phase of the Spiral Model, the features of the product dated and analyzed, and the risks at that point in time are identified and are resolved through prototyping. Thus, this model is much more flexible compared to other SDLC models.

**Why Spiral Model is called Meta Model?**  
The Spiral model is called a Meta-Model because it subsumes all the other SDLC models. For example, a single loop spiral actually represents the iterative waterfall model The spiral model incorporates the stepwise approach of the Classical Waterfall Model. The spiral model uses the approach of the Prototyping Model by building a prototype at the start of each phase as a risk-handling technique. Also, the spiral model can be considered as supporting the Evolutionary model– the iterations along the spiral can be considered as evolutionary levels through which the complete system is built.

**Advantages of Spiral Model**:   
Below are some advantages of the Spiral Model.

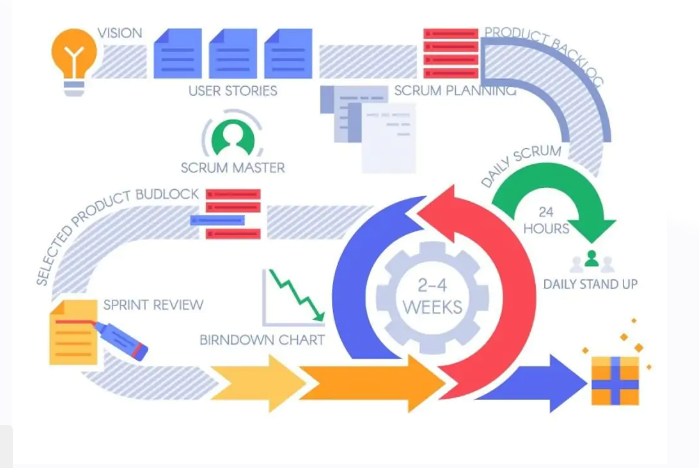
1. **Risk Handling:** The projects with many unknown risks that occur as the development proceeds, in that case, Spiral Model is the best development model to follow due to the risk analysis and risk handling at every phase.
2. **Good for large projects:** It is recommended to use the Spiral Model in large and complex projects.
3. **Flexibility in Requirements:** Change requests in the Requirements at later phase can be incorporated accurately by using this model.
4. **Customer Satisfaction:** Customer can see the development of the product at the early phase of the software development and thus, they habituated with the system by using it before completion of the total product.

**Disadvantages of Spiral Model**:   
Below are some main disadvantages of the spiral model.

1. **Complex:** The Spiral Model is much more complex than other SDLC models.
2. **Expensive:** Spiral Model is not suitable for small projects as it is expensive.
3. **Too much dependability on Risk Analysis:** The successful completion of the project is very much dependent on Risk Analysis. Without very highly experienced experts, it is going to be a failure to develop a project using this model.
4. **Difficulty in time management:** As the number of phases is unknown at the start of the project, so time estimation is very difficult.

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.



Scrum is precisely an evolution of Agile Management. Scrum methodology is based on a set of very defined practices and roles that must be involved during the software development process. It is a flexible methodology that rewards the application of the 12 agile principles in a context agreed by all the team members of the product.

Scrum is executed in temporary blocks that are short and periodic, called Sprints, which usually range from 2 to 4 weeks, which is the term for feedback and reflection. Each Sprint is an entity in itself, that is, it provides a complete result, a variation of the final product that must be able to be delivered to the client with the least possible effort when requested.

The process has as a starting point, a list of objectives/ requirements that make up the project plan. It is the client of the project that prioritizes these objectives considering a balance of the value and the cost thereof, that is how the iterations and consequent deliveries are determined.

On the one hand the market demands quality, fast delivery at lower costs, for which a company must be very agile and flexible in the development of products, to achieve short development cycles that can meet the demand of customers without undermining the quality of the result. It is a very easy methodology to implement and very popular for the quick results it gets.

Scrum methodology is used mainly for software development, but other sectors are also taking advantage of its benefits by implementing this methodology in their organizational models such as sales, marketing, & HR teams etc.

Benefits of Scrum Methodology

Scrum has many advantages over other agile development methodologies. It is currently the most used and trusted framework of reference in the software industry. Below are some of the known benefits of Scrum:

Easily Scalable: Scrum processes are iterative and are handled within specific work periods, which makes it easier for the team to focus on definite functionalities for each period. This not only has the benefit of achieving better deliverables in line with the needs of the user, but also gives the ability to the teams to scale the modules in terms of functionality, design, scope and characteristics in an orderly, transparent and simple manner.

Compliance of expectations: The client establishes their expectations indicating the value that each requirement/ history of the project brings, the team estimates them and with this information the Product Owner establishes its priority. On a regular basis, in the sprint demos, the Product Owner verifies that the requirements have been met and transmits feedback to the team.

Flexible to changes: Quick reaction to changes in requirements generated by customer needs or market developments. The methodology is designed to adapt to the changing requirements that complex projects entail.

Time to Market reduction: The client can start using the most important functionalities of the project before the product is completely ready.

Higher software quality: The working method and the need to obtain a functional version after each iteration, helps to obtain a higher quality software.

Timely Prediction:  Using this methodology, we know the average speed of the team by sprint (story points), with which, consequently, it is possible to estimate when a certain functionality that is still in the backlog will be available.

Reduction of risks:  The fact of carrying out the most valuable functionalities in the first place and of knowing the speed with which the team advances in the project, allows to clear risks effectively in advance.

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.

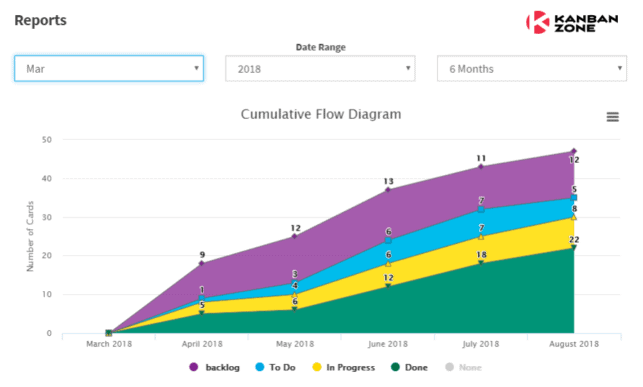
Daily Scrum: The objective of the Daily Scrum is to evaluate the progress and trend until the end of the Sprint, synchronizing the activities and creating a plan for the next 24 hours. It is a brief meeting that takes place daily during the Sprint period. Three questions are answered individually:  What 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.

5. Explain the utility of Kanban CFD reports.

A Cumulative Flow Diagram is a graphical representation of work as it flows through your Kanban system. It is a time-based plot, with the time interval in the x-axis and the number of cards in the y-axis. The graph is divided into different colored bands, with the bands representing a state or column in your Kanban board.



## How to Read a CFD Chart

The topmost band in a cumulative flow diagram represents the items that your team needs to do, while the bottom band represents the items that your team has done. The bands in between represent the items that your team is currently working on or the in progress items. In our example above, the Backlog and To Do bands represent the To Do states. The graph generated can vary depending on the configuration of your CFD chart tool if you are using an application.

The Cumulative Flow Diagram shows how many items have moved from one state to another in a given period. Depending on the tool you’re using, it should be able to give you the actual count of items per band per time interval. You can determine how many items are still waiting to be done and how many items you’re currently working on.

The trajectory of the chart should be consistently upwards, with the bands staying more or less parallel and even in width. This means that you are delivering work at a stable pace. The exception is the Done band since it should widen over time indicating more items have been completed. But not all projects are perfect and there are times when an upwards slope is not achieved. Those times can be indications of possible issues within your workstream. The width of your bands can also indicate problems in your flow.

Here are some of the common patterns that you need to watch out for:

* **Narrowing Bands** – This indicates that there is lesser work than the team’s capacity. There may be a need to reallocate team members to balance out the workload.
* **Widening Bands** – Indicates a bottleneck. This means that the entry rate of the items to this state is faster than its exit rate. Analyze what is causing the holdup and apply corrective action.
* **To Do band is wider than your Done band** – Indicates that your team cannot consume the work as fast as you are adding them. This can mean several things which can include capacity problems and insufficient information about the tasks.
* **Slope is flat** – Indicates that no work has been done. Not always a cause for alarm as this can happen during holidays or no workdays. But if this isn’t the case then you need to dig into why nothing’s getting done. Is there a problem with your production servers? Are your team members missing?

## How to Use CFD to Improve Team Performance

The cumulative flow diagram provides insights that may not be apparent with just basing progress on a Kanban board. As the CFD chart includes historical data, it shows trends and patterns that depict the performance of the team through a given period. With a single view, one can be able to spot if a bottleneck is becoming a recurring problem within a process state. If there are deviations to what should be the trajectory or form of the chart, the team can apply corrective action immediately.

Since the CFD chart also shows performance metrics such as WIP, lead time, and cycle time, the team can use these data to estimate when they are likely to complete a project. If given a target release date, the team can use their historical performance on assessing whether they can meet it or not. The chart can also be used by management as a basis for capacity planning needs. Overall, the cumulative flow diagram is a very useful tool to measure the performance of a team and create a stable and predictable flow.

## How to Get a Cumulative Flow Diagram Tool

If you’re using a Kanban tool, like Kanban Zone, you should be able to use a built-in CFD report generator within the application. With an online Kanban tool that has rich report features, you don’t need to do the analytics yourself. But if you want to get your hands on the data, you can do this manually through a spreadsheet and create the chart from scratch.

What you need to ensure though is that your team is making timely updates to your cards and moving them to their appropriate states. Else, your CFD chart will not be an accurate representation of your workflow.