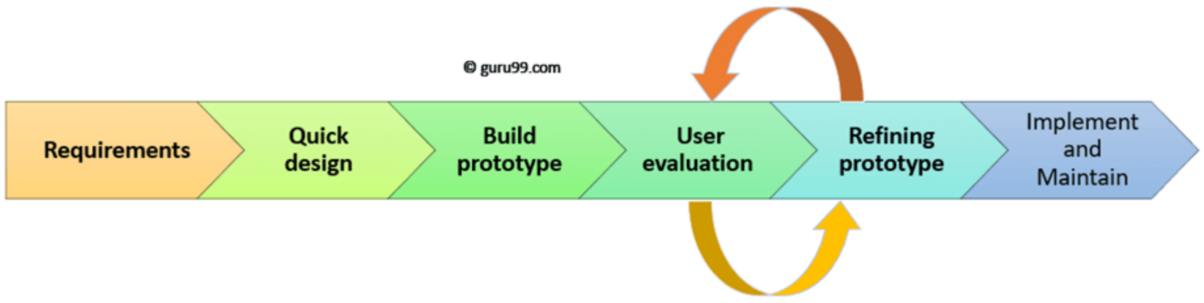
Q1)Discuss the prototyping model. What is the effect of designing a prototype on the overall cost of the project?
What does prototyping mean?
<u>Ans:</u>

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 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 reframing it after its full-fledged development- which might cost a fortune.

Prototyping Model Phases



Prototyping Model has following six SDLC phases as follow:

Step 1: Requirements gathering and analysis

A prototyping model starts with requirement analysis. In this phase, the requirements of the system are defined in detail. During the process, the users of the system are interviewed to know what is their expectation from the system.

Step 2: Quick design

Step 3: Build a Prototype

In this phase, an actual prototype is designed based on the information gathered from quick design. It is a small working model of the required system.

Step 4: Initial user evaluation

In this stage, the proposed system is presented to the client for an initial evaluation. It helps to find out the strength and weakness of the working model. Comment and suggestion are collected from the customer and provided to the developer.

Step 5: Refining prototype

If the user is not happy with the current prototype, you need to refine the prototype according to the user's feedback and suggestions.

This phase will not over until all the requirements specified by the user are met. Once the user is satisfied with the developed prototype, a final system is developed based on the approved final prototype.

Step 6: Implement Product and Maintain

Once the final system is developed based on the final prototype, it is thoroughly tested and deployed to production. The system undergoes routine maintenance for minimizing downtime and prevent large-scale failures.

Types of Prototyping Models

Four types of Prototyping models are:

1. Rapid Throwaway prototypes
2. Evolutionary prototype
3. Incremental prototype
4. Extreme prototype

Rapid Throwaway Prototype

Advantages of Prototyping Model:

- **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.

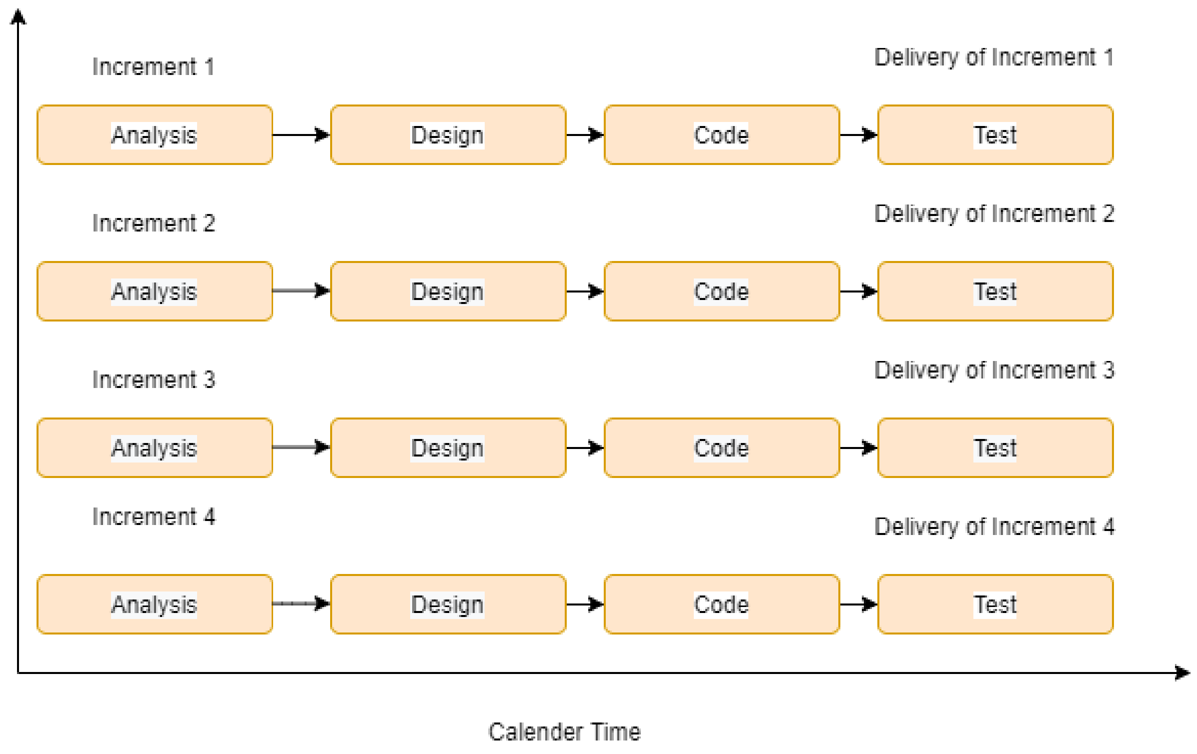
- **Communicate your ideas for meeting the client's business objectives**– It helps to communicate the idea if it does not exactly meet the client's business objectives, and you can avoid the complete restructuring of the developed product by just revising the product prototype.
- **Coming up with solutions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

Q2)Compare iterative enhancement model and evolutionary process model.

Ans:

Iterative Enhancement Model:

The iterative enhancement model in software engineering combines elements of the linear sequential model with the iterative philosophy of prototyping. In this model, the software is broken down into several modules which are incrementally developed and delivered. Firstly, the development team develops the core module of the system. After that, it is refined into increasing levels of capacity of adding new functionalities in successive versions.



When it is used the first increment is often a core product i.e basic requirements are addressed but many supplementary features are undelivered. The core product is used by the customers. As a result of use or evaluation, a plan is developed for the next increment. The plan addresses the modification of the core product to meet the needs of customers, and the delivery of additional features and functionality. This process is repeated following each delivery of increment until the complete product is produced.

Advantages of Iterative Enhancement Model:

- The feedback from early increment improves the later stages.
- The possibility of changes in requirement is reduced due to the shorter time span between the design component and its delivery.
- Users get benefits earlier than the conventional approach.
- Smaller sub-projects are easier to control and manages.
- File priority features are incorporated in early deliverables.

Diadvantage of Model:

- Programmers may have more productive working on one large system than on the series.
- The development cost of the total product is higher.
- The time period for the delivery of the entire functionality is higher.
- Planning of delivery increments is critical to success.
- Wrong planning results to diaster.

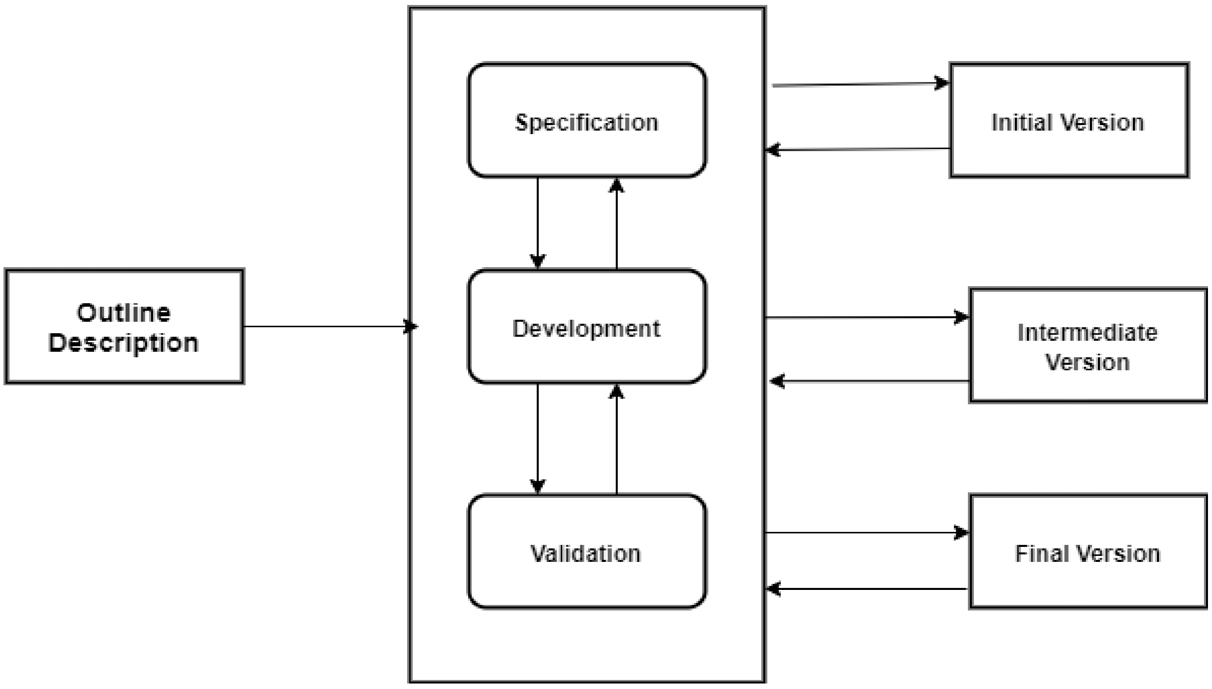
When to Use Iterative Model:

- Model is used where risk of long project can not be taken.
- This model is for businesses where time is of essence.

The evolutionary model is an iterative model as they are characterized in a manner that enables software engineering to develop a complete version of the software. Such models are applied because the requirements often change. So, the end product will be unrealistic, for a complete version is impossible due to tight market deadlines. It is better to introduce a limited version. Thus, Software engineering can follow a process model that has been explicitly designed to accommodate a product that gradually competes over time.

Evolutionary Development Model

Evolutionary model is a combination of Iterative and Incremental 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 Development Model

Advantages of the Model:

- Risk analysis is better.
- It supports changing environment.
- Initial operating time is less.
- Better suited for large mission-critical projects.
- During the life cycle software is produced early which facilitates customer evaluation and feedxback.

Disadvantage of the Model:

- Management complexity is more.
- Not suitable for smaller projects.
- Can be costly to use.
- Highly skilled resources are required for risk analysis.

Where to use Evolutionary Model:

- Very useful for large projects.
- Well suited for projects using object-oriented development.
- When a client prefers to have the product in increments so that he can began using the specific characteristics as they are provided rather than waiting for the entire thing to be manufactured and delivered, this model is frequently adopted

.

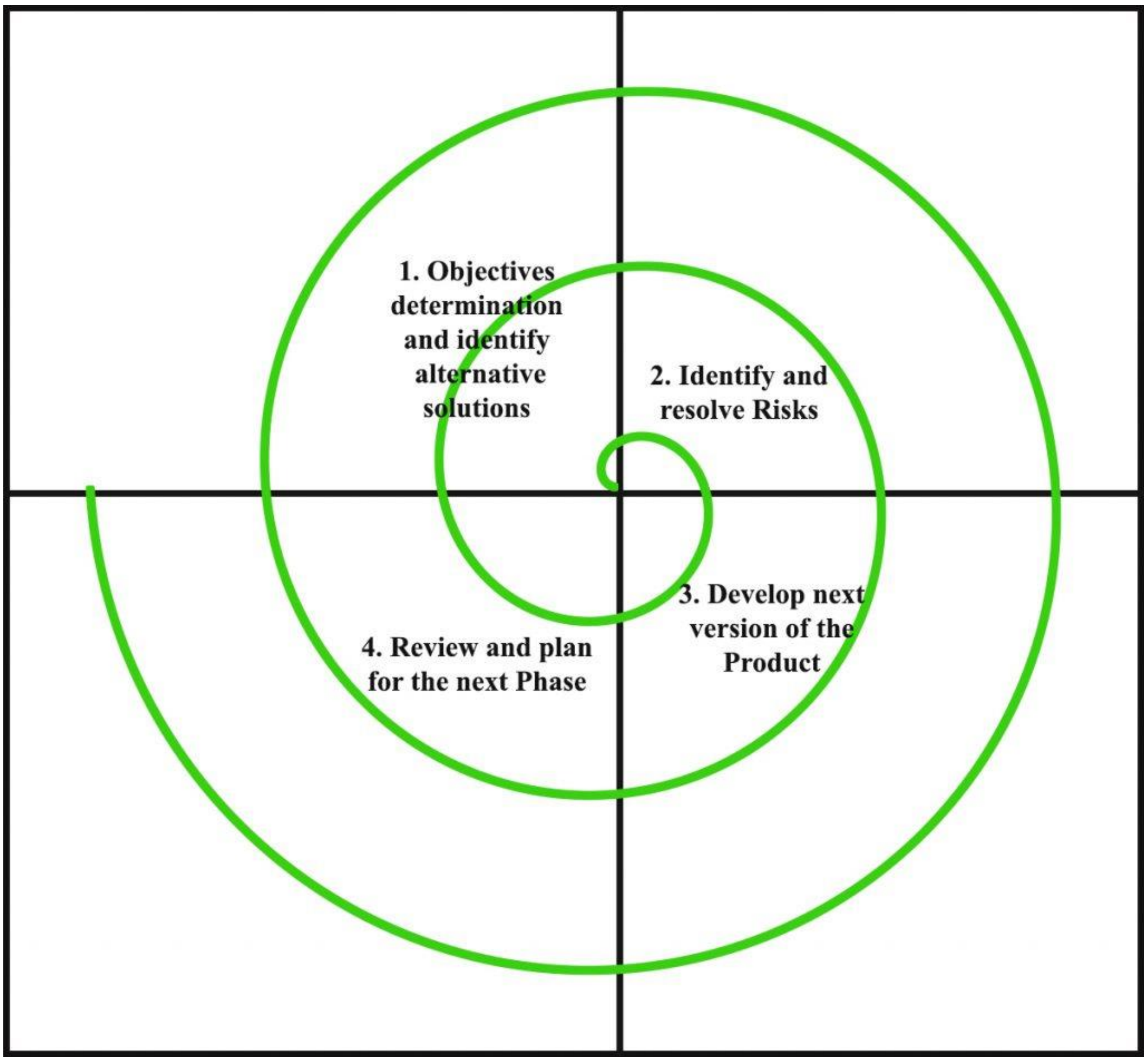
Q3)As we move outward along with process flow path ofthe spiral model, what can we say about software

Ans:

As work moves outward on the spiral, the product moves toward a more complete state and the level of abstraction at which work is performed is reduced (i.e., implementation specific work accelerates as we move further from the origin).

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: –


1. Objectives determination and identify alternative solutions
2. Identify and resolve Risks
3. Develop next version of the Product
4. Review and plan for the next Phase

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.

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.
5.

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.

Ans:

What is Agile Scrum Methodology?

Scrum is a framework that was created as a part of the Agile methodology. It was made to help teams address complex problems within a project. The Scrum method was created to improve productivity within the team and deliver products that deliver the highest value to the consumer in a creative manner.

The Agile Scrum Methodology is focused on improving teamwork through continuous learning using iterations. Scrum was mainly used by software development teams but has now grown to be accepted within other industries as well. The framework has principles and lessons that can be adapted by any team in any industry, making it all the more popular. The Scrum method consists of various tools, roles, and techniques that team members can learn to help them improve the structure of how they work.

Scrum, when applied correctly can be very successful. More than 62% of the projects completed using scrum have been successful. An ideal Scrum team is usually made of between four to nine team members, which is what leads to the highest levels of success.

There are three pillars of empiricism that the Scrum method upholds, which are listed below.

Transparency

Transparency is important for all stakeholders. There are certainly important aspects of the development process that need to be visible at all times to the team so that there is a clear outcome. These aspects need to be well-defined and standardized so everyone can easily understand and interpret all the information.

Inspection

All the artifacts used in every sprint need to be thoroughly examined by the Scrum team. This inspection helps in detecting any disruptions or undesirable variances in the process. It needs to be done regularly but in a way that it does not interrupt the sprint.

Adaptation

If there are any issues found during the inspection, the problems need to be addressed. This is so that the end product is acceptable to the customers. It can be solved by adapting or making adjustments to the processes so that all future deviations are minimized.

The Scrum Artifacts are crucial aspects of the Agile Scrum Methodology to help with the product development process. They give the Scrum team all the important information necessary to create a successful sprint. The artifacts consist of Product Backlog, Sprint Backlog, and Increment.

Product Backlog

A product backlog is essentially a list of requirements for a product. It is made up of all the features, functions, enhancements, and fixes that are necessary for a product life cycle. According to a recent survey, 70% of the PBIs take less than three weeks to go through the entire workflow of all product backlogs. A product backlog is updated regularly based on the change in requirements.

Sprint Backlog

The PBIs are divided into sprints, creating the Sprint Backlog, which is essentially the list of the items in a product backlog that need to be completed within a sprint. A sprint backlog also includes a plan to deliver product increments and how the team can achieve the Sprint Goal. It defines the work of the development team that is necessary to meet the objective of the sprint.

Increment

The final artifact is an increment, which is the total of all the completed product backlog items within the duration of a sprint and the sprints before. After each sprint, the increment needs to be marked as "Done," which is an acceptable usable condition for the product, whether or not it gets released.

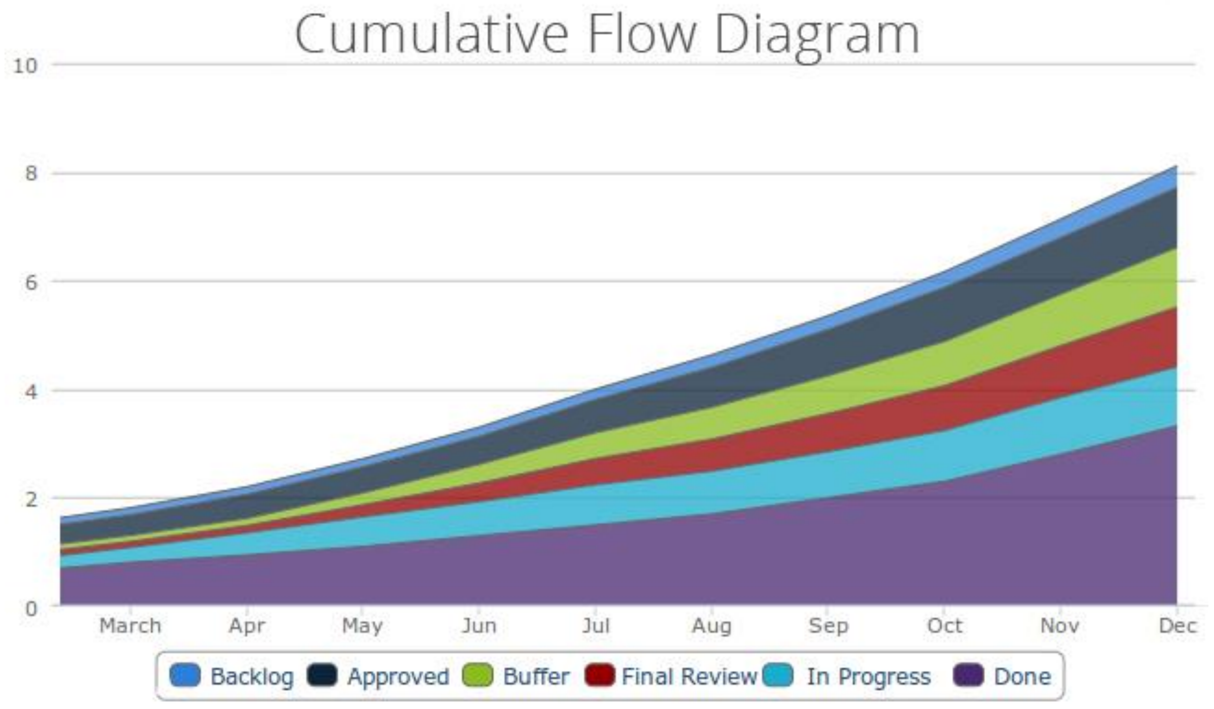
Q5)Explain the utility of Kanban CFD reports.

Ans:

Cumulative Flow Diagram is an analytical tool, fundamental to Kanban method. It allows teams to visualize their effort and project progress. When there's an impediment about to occur within the process - the CFD is where you'll see it first. Instead of the graph staying smooth and rising gently, there will be a bump, a sudden ascend or descend. So, where being able to predict problems is concerned, this is the very graph you need.

The Cumulative Flow Diagram visualises how tasks mount up over time, together with their distribution along the process stages. The graph is built from different colored bands of tasks gathered in various columns. One color represents one column - so that each band shows how many tasks sit at what stage of the process, in a given time - the horizontal value.

The ideal diagram you want to see is an evenly rising one, with bands staying more-less even, except for the "completed tasks" band, which should continuously be getting taller, just as the number of done tasks is hopefully always getting higher.



The CFD only requires 3 basic things from the process - a Backlog, an In Progress column and a Done section - using this type of division allows you to read valid and usable information from the diagram. Therefore, any team, that utilizes this kind of workflow division, can benefit from Cumulative Flow. Whether you use **Scrum**, **Kanban** or any other custom project management method, for as long as you organize it in task groups, the CFD will be of great help.

Submitted By: Mayur Rajendra Lokhande.