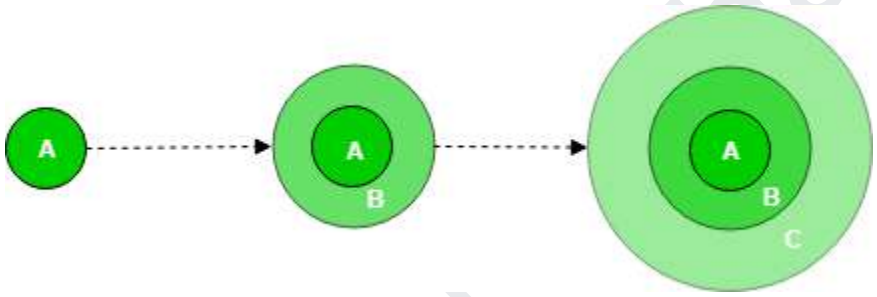
❖ Incremental Process Models

The incremental process model is also known as the Successive version model. First, a simple working system implementing only a few basic features is built and then that is delivered to the customer. Then thereafter many successive iterations/ versions are implemented and delivered to the customer until the desired system is released.



A, B, and C are modules of Software Products that are incrementally developed and delivered.

Life Cycle Activities

Requirements of Software are first broken down into several modules that can be incrementally constructed and delivered.

At any time, the plan is made just for the next increment and not for any kind of long-term plan. Therefore, it is easier to modify the version as per the need of the customer.

The Development Team first undertakes to develop core features (these do not need services from other features) of the system.

Once the core features are fully developed, then these are refined to increase levels of capabilities by adding new functions in Successive versions.

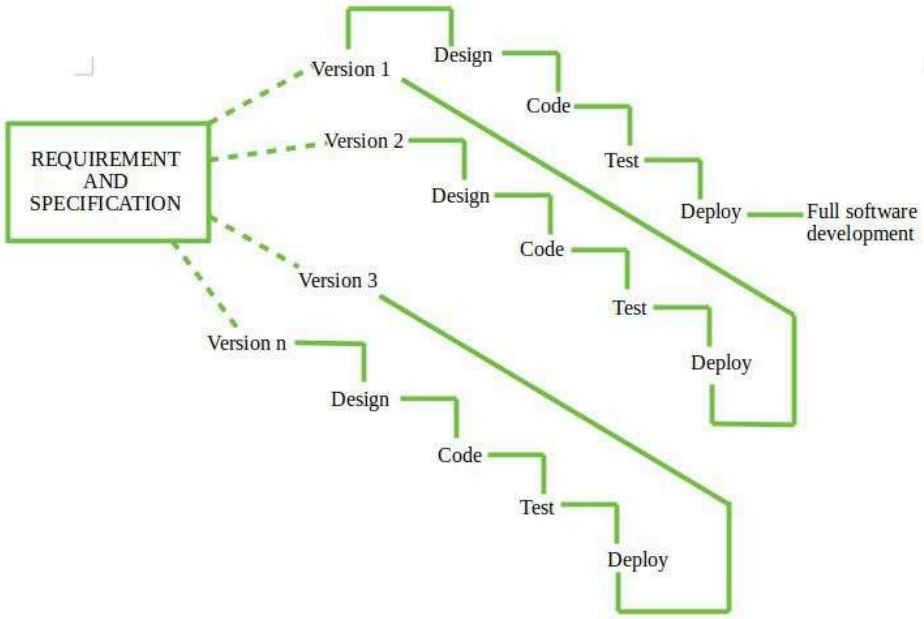
Each incremental version is usually developed using an iterative waterfall model of development.

As each successive version of the software is constructed and delivered, now the feedback of the Customer is to be taken and these were then incorporated into the next version.

Each version of the software has more additional features than the previous ones.

After Requirements gathering and specification, requirements are then split into several different versions starting with version 1, in each successive increment, the next version is constructed and then deployed at the customer site.

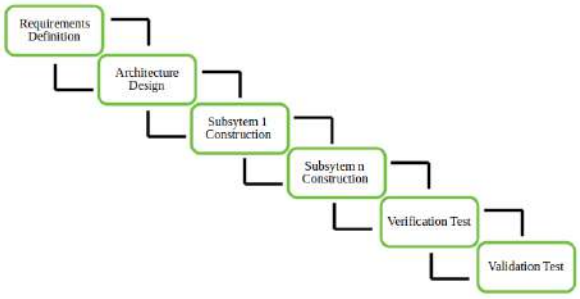
After the last version (version n), it is now deployed at the client site.



Types of Incremental Model

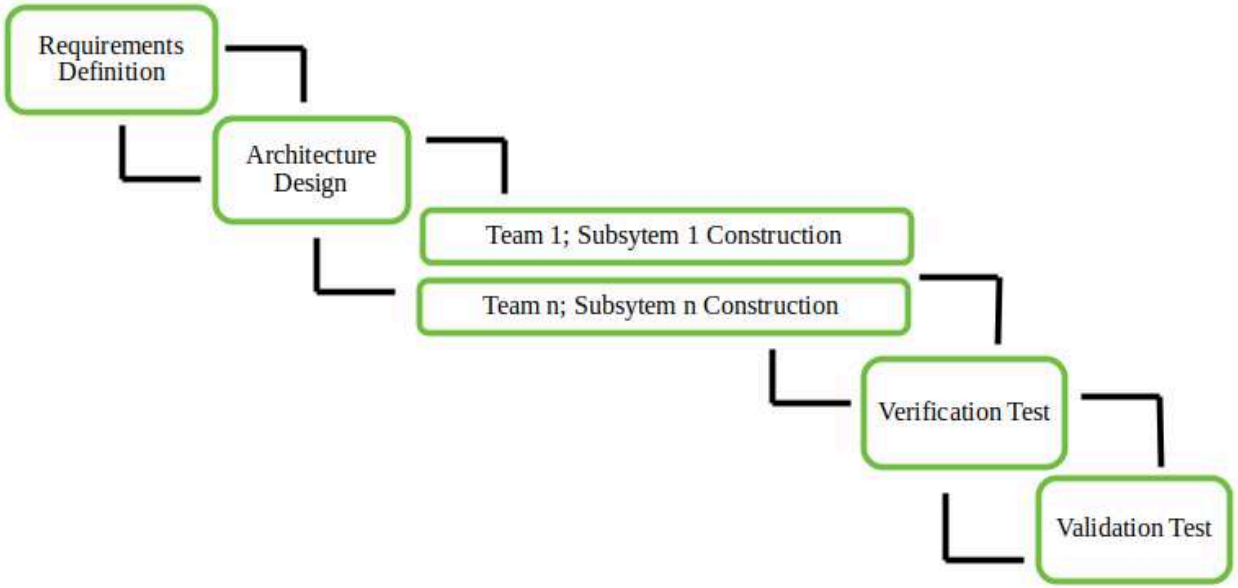
1. Staged Delivery Model

Construction of only one part of the project at a time.



2. Parallel Development Model

Different subsystems are developed at the same time. It can decrease the calendar time needed for the development, i.e. TTM (Time to Market) if enough resources are available.



When to use Incremental Process Model

- Funding Schedule, Risk, Program Complexity, or need for early realization of benefits.
- When Requirements are known up-front.
- When Projects have lengthy development schedules.
- Projects with new Technology.
- Error Reduction (core modules are used by the customer from the beginning of the phase and then these are tested thoroughly).

- Uses divide and conquer for a breakdown of tasks.
- Lowers initial delivery cost.
- Incremental Resource Deployment.
- Requires good planning and design.
- The total cost is not lower.
- Well-defined module interfaces are required.

Characteristics of Incremental Process Model

- System development is divided into several smaller projects.
- To create a final complete system, partial systems are constructed one after the other.
- Priority requirements are addressed first.
- The requirements for that increment are frozen once they are created.

Advantages of Incremental Process Model

- Prepares the software fast.
- Clients have a clear idea of the project.
- Changes are easy to implement.
- Provides risk handling support, because of its iterations.
- Adjusting the criteria and scope is flexible and less costly.
- Comparing this model to others, it is less expensive.
- The identification of errors is simple.

Disadvantages of Incremental Process Model

- A good team and proper planned execution are required.
- Because of its continuous iterations the cost increases.

- Issues may arise from the system design if all needs are not gathered upfront throughout the duration of the program lifecycle.
- Every iteration step is distinct and does not flow into the next.
- It takes a lot of time and effort to fix an issue in one unit if it needs to be corrected in all the units.

❖ Evolutionary Process Models

Evolutionary models are iterative type models. They allow developers to develop more complete versions of the software.

Following are the evolutionary process models.

1. The prototyping model

2. The spiral model

The prototyping model

Prototype is defined as the first or preliminary form using which other forms are copied or derived.

Prototype model is a set of general objectives for software.

It does not identify the requirements like detailed input, output.

It is software working model of limited functionality

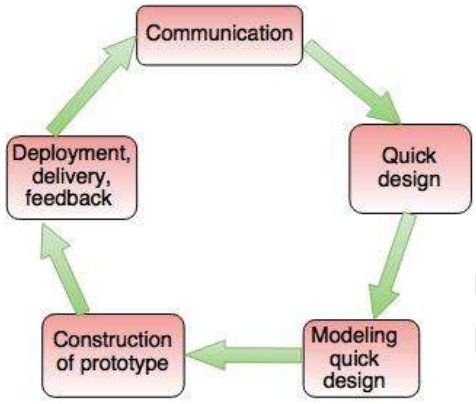


Fig. - The Prototyping Model

The different phases of Prototyping model are:

1. Communication

In this phase, developers and customers meet and discuss the overall objectives of the software.

2. Quick design

Quick design is implemented when requirements are known.

It includes only the important aspects like input and output format of the software.

It focuses on those aspects which are visible to the user rather than the detailed plan.

It helps to construct a prototype.

3. Modeling quick design

This phase gives the clear idea about the development of software

It allows the developer to better understand the exact requirements.

4. Construction of prototype

The prototype is evaluated by the customer itself.

5. Deployment, delivery, feedback

If the user is not satisfied with the current prototype then it refines according to the requirements of the user.

The process of refining the prototype is repeated until all the requirements of users are met.

When the users are satisfied with the developed prototype then the system is developed on the basis of the final prototype.

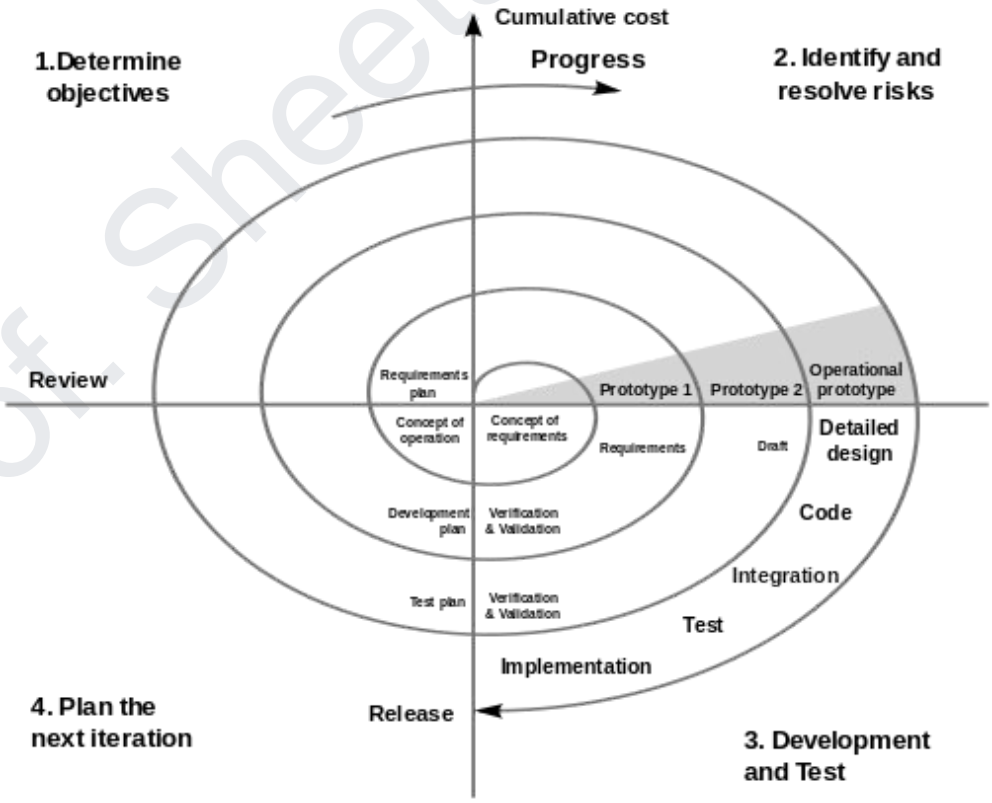
The spiral model

It provides a systematic and iterative approach to software development. In its diagrammatic representation,

looks like a spiral with many loops.

The exact number of loops of the spiral is unknown and can vary from project to project.

It is based on the idea of a spiral, with each iteration of the spiral representing a complete software development cycle, from requirements gathering and analysis to design, implementation, testing, and maintenance.



Phases:

1. Determine Objective

The first phase of the Spiral Model where the scope of the project is determined

2. Risk Analysis

In the risk analysis phase, the risks associated with the project are identified and evaluated.

3. Engineering

In the engineering phase, the software is developed based on the requirements gathered in the previous iteration.

4. Evaluation

In the evaluation phase, the software is evaluated to determine if it meets the customer's requirements and if it is of high quality.

5. Planning

The next iteration of the spiral begins with a new planning phase, based on the results of the evaluation.

The Spiral Model is often used for complex and large software development projects, as it allows for a more flexible and adaptable approach to software development.

It is also well-suited to projects with significant uncertainty or high levels of risk.

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

Advantages of the Spiral Model

Below are some advantages of the Spiral Model.

- 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.
- Good for large projects: It is recommended to use the Spiral Model in large and complex projects.

- Flexibility in Requirements: Change requests in the Requirements at a later phase can be incorporated accurately by using this model.
- Customer Satisfaction: Customers 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.
- Iterative and Incremental Approach: The Spiral Model provides an iterative and incremental approach to software development, allowing for flexibility and adaptability in response to changing requirements or unexpected events.
- Emphasis on Risk Management: The Spiral Model places a strong emphasis on risk management, which helps to minimize the impact of uncertainty and risk on the software development process.
- Improved Communication: The Spiral Model provides for regular evaluations and reviews, which can improve communication between the customer and the development team.
- Improved Quality: The Spiral Model allows for multiple iterations of the software development process, which can result in improved software quality and reliability.

Disadvantages of the Spiral Model

Below are some main disadvantages of the spiral model.

- Complex: The Spiral Model is much more complex than other SDLC models.
- Expensive: Spiral Model is not suitable for small projects as it is expensive.
- 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.
- Difficulty in time management: As the number of phases is unknown at the start of the project, time estimation is very difficult.
- Complexity: The Spiral Model can be complex, as it involves multiple iterations of the software development process.

- Time-Consuming: The Spiral Model can be time-consuming, as it requires multiple evaluations and reviews.
- Resource Intensive: The Spiral Model can be resource-intensive, as it requires a significant investment in planning, risk analysis, and evaluations.

❖ Concurrent Models

The concurrent development model is called a concurrent model.

The communication activity has completed in the first iteration and exits in the awaiting changes state.

The modeling activity completed its initial communication and then went to the underdevelopment state.

If the customer specifies the change in the requirement, then the modeling activity moves from the under development state into the awaiting change state.

The concurrent process models activities moving from one state to another state.

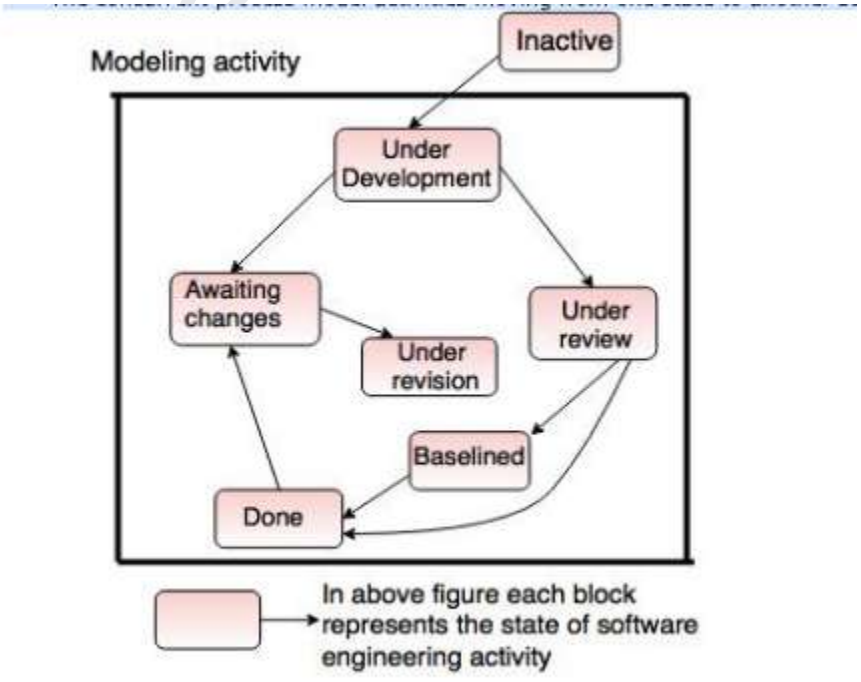


Fig. - One element of the concurrent process model

Advantages of the concurrent development model

- This model is applicable to all types of software development processes.
- It is easy to understand and use.
- It gives immediate feedback from testing.
- It provides an accurate picture of the current state of a project

Disadvantages of the concurrent development model

- It needs better communication between the team members. This may not be achieved all the time.
- It requires us to remember the status of the different activities.

❖ Agile process, Agility Principles

In earlier days, the Iterative Waterfall Model was very popular for completing a project.

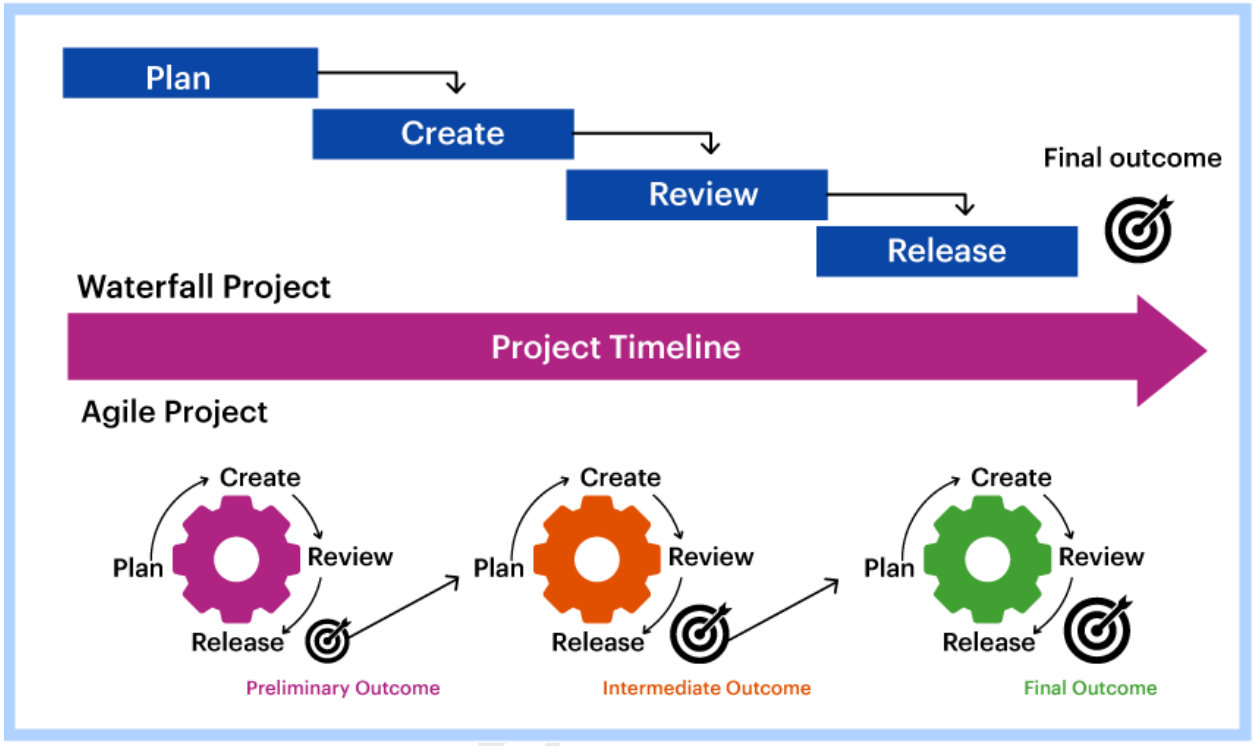
But nowadays, developers face various problems while using it to develop software.

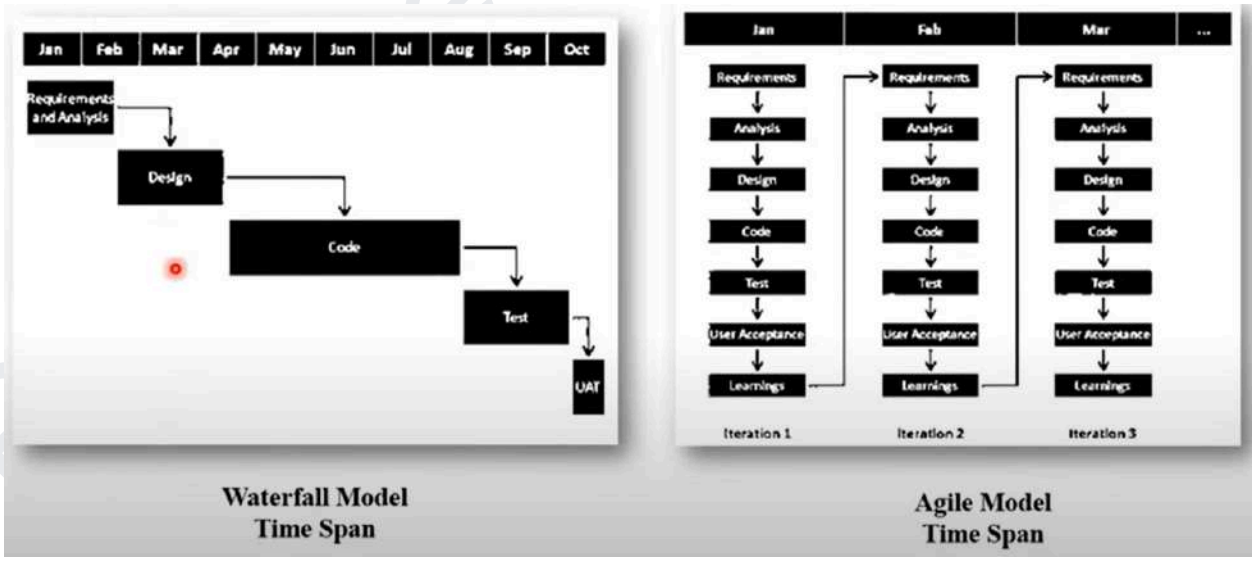
The main difficulties included handling changes in requirements during project development and the high cost and time required to incorporate these changes.

To overcome these drawbacks of the Waterfall Model, in the mid-1990s the Agile Software Development model was proposed.

Agile Software Development is widely used by software development teams and is considered to be a flexible and adaptable approach to software development that is well-suited to changing requirements and the fast pace of software development.

Agile is a time-bound, iterative approach to software delivery that builds software incrementally from the start of the project, instead of trying to deliver all at once.





Principles of Agile:

- The highest priority is to satisfy the customer through early and continuous delivery of valuable software.

- It welcomes changing requirements, even late in development.
- Deliver working software frequently, from a couple of weeks to a couple of months, with a preference for the shortest timescale.
- Build projects around motivated individuals. Give them the environment and the support they need and trust them to get the job done.
- Working software is the primary measure of progress.
- Simplicity the art of maximizing the amount of work not done is essential.
- The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
- By the amount of work that has been finished, gauge your progress.
- Never give up on excellence.
- Take advantage of change to gain a competitive edge.

The Agile Software Development Process

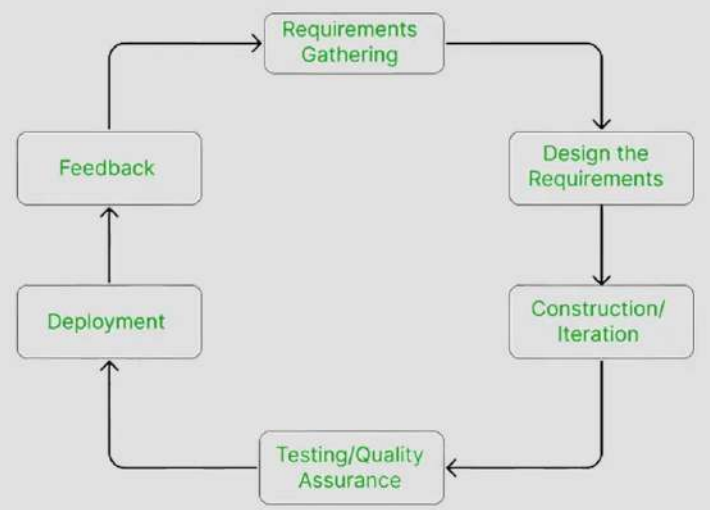


Fig:- Steps in Agile SDLC Model

1.Requirement Gathering:- In this step, the development team must gather the requirements, by interaction with the customer. The development team should plan the time and effort needed to build the project. Based on this information you can evaluate technical and economical feasibility.

2.Design the Requirements:- In this step, the development team will use user-flow-diagram or high-level UML diagrams to show the working of the new features and show how they will apply to the existing software.

3. Construction / Iteration:- In this step, development team members start working on their project, which aims to deploy a working product.

4. Testing / Quality Assurance:- Testing involves Unit Testing, Integration Testing, and System Testing. A brief introduction of these three tests is as follows:

Unit Testing:- Unit testing is the process of checking small pieces of code to ensure that the individual parts of a program work properly on their own. Unit testing is used to test individual blocks (units) of code.

Integration Testing:- Integration testing is used to identify and resolve any issues that may arise when different units of the software are combined.

System Testing:- Goal is to ensure that the software meets the requirements of the users and that it works correctly in all possible scenarios.

5. Deployment:- In this step, the development team will deploy the working project to end users.

6. Feedback:- This is the last step of the Agile Model. In this, the team receives feedback about the product and works on correcting bugs based on feedback provided by the customer.

Advantages:

It reduces the total development time of the whole project.

Agile development emphasizes face-to-face communication among team members, leading to better collaboration and understanding of project goals.

Customer representatives get the idea of updated software products after each iteration. So, it is easy for him to change any requirement if needed.

Agile development puts the customer at the center of the development process, ensuring that the end product meets their needs.

Disadvantages:

The agile model depends highly on customer interactions so if the customer is not clear, then the development team can be driven in the wrong direction.

Agile development models require a high degree of expertise from team members, as they need to be able to adapt to changing requirements and work in an iterative environment. This can be challenging for teams that are not experienced in agile development practices and can lead to delays and difficulties in the project.

Due to the absence of proper documentation, when the project completes and the developers are assigned to another project, maintenance of the developed project can become a problem.

❖ Extreme Programming (XP)

Extreme programming (XP) is one of the most important software development frameworks of Agile models.

It is used to improve software quality and responsiveness to customer requirements.

The extreme programming model recommends taking the best practices that have worked well in the past in program development projects to extreme levels.

A developer focuses on the framework activities like planning, design, coding and testing.

XP has a set of rules and practices.

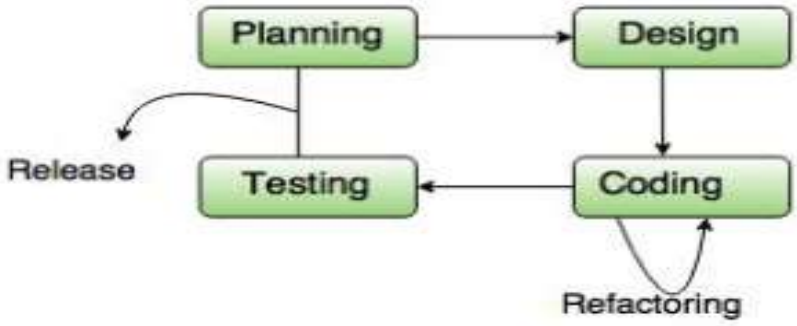


Fig. - The Extreme Programming Process

XP Values: Values offer teams purpose, guiding them in making high-level decisions

Communication: You can't work together effectively without sharing knowledge, and you can't share knowledge without communication.

Simplicity: Developers can save time and effort by writing simple, effective code that works properly. Ultimately, the less complex code enhances product value.

Feedback: Early, constant feedback is ideal for team members who release frequent software deliveries, helping them to adjust as the project evolves and changes. The sooner programmers know that the product requires changes, the easier it is to create those changes (and less painful).

Respect: Every team member cares about their work, and everyone contributes.

Courage: It takes guts to admit you're mistaken and that your idea didn't work and must be changed. Being honest takes courage, and you need honesty in providing realistic estimates, even if stakeholders don't like the truth.

XP Process: The XP process comprises four framework activities

Planning

Begins with the listening, leads to creation of "user stories" that describes required output, features, and functionality.

Customer assigns a value(i.e., a priority) to each story.

The Agile team assesses each story and assigns a cost (development weeks. If more than 3 weeks, customer asked to split into smaller stories)

Design

begin with the simplest possible design, knowing that later iterations will make them more complex

Coding

Recommends the construction of a unit test for a story before coding commences. So implementer can focus on what must be implemented to pass the test.

Encourages "pair programming". Two people work together at one workstation.

Testing

Validation testing of the system occurs on a daily basis. It gives the XP team a regular indication of the progress.

'XP acceptance tests' are known as the customer test

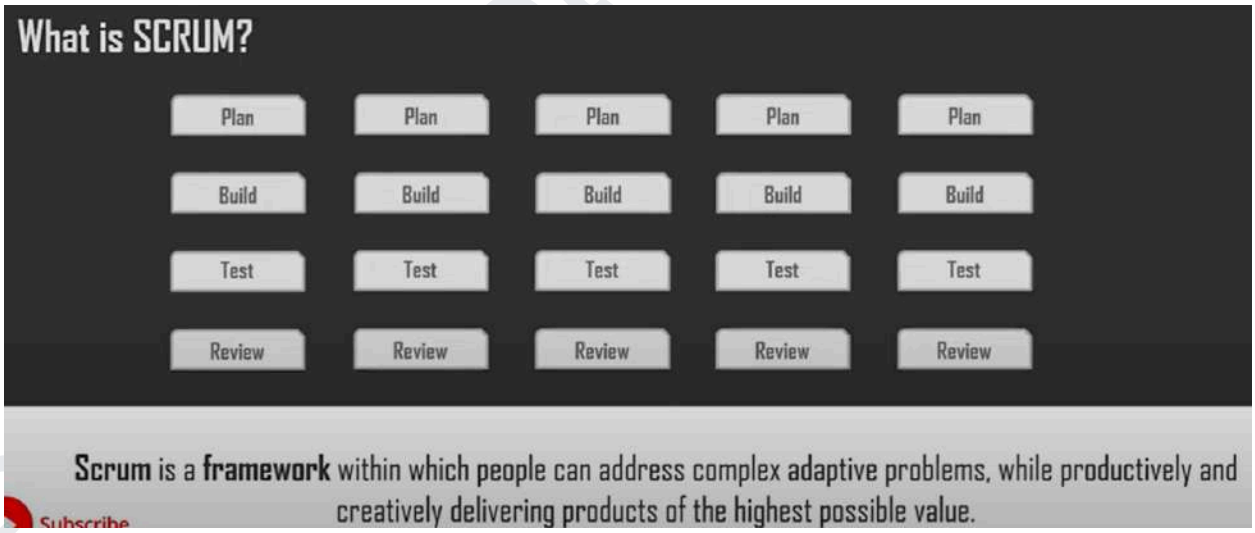
❖ Scrum

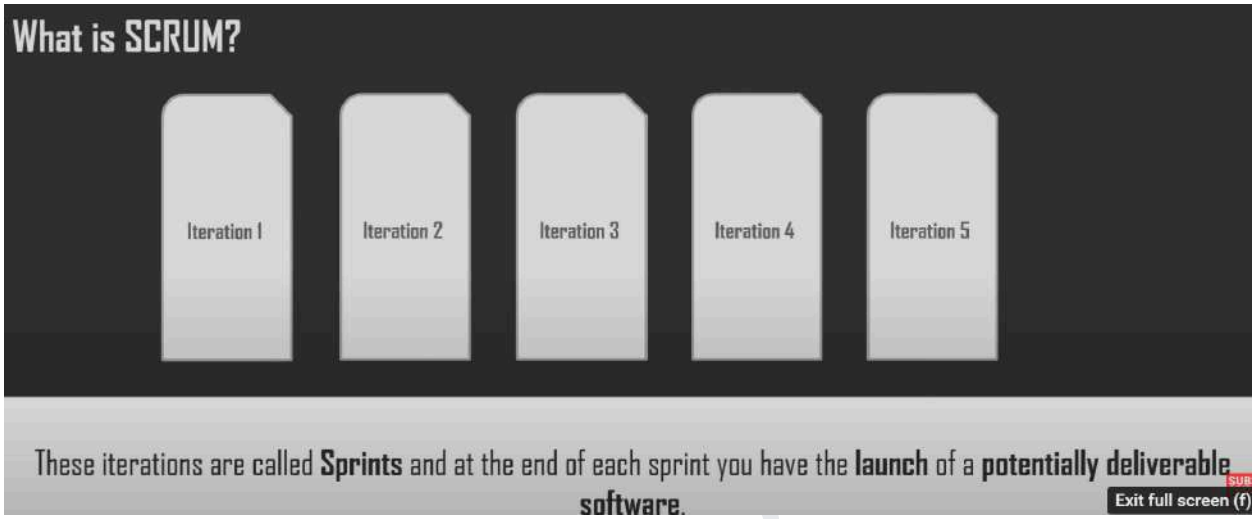
Scrum is the type of Agile framework. It is a framework within which people can address complex adaptive problems while productivity and creativity of delivering product is at highest possible values. Scrum uses an Iterative process. Salient features of Scrum are:

Scrum emphasizes self-organization.

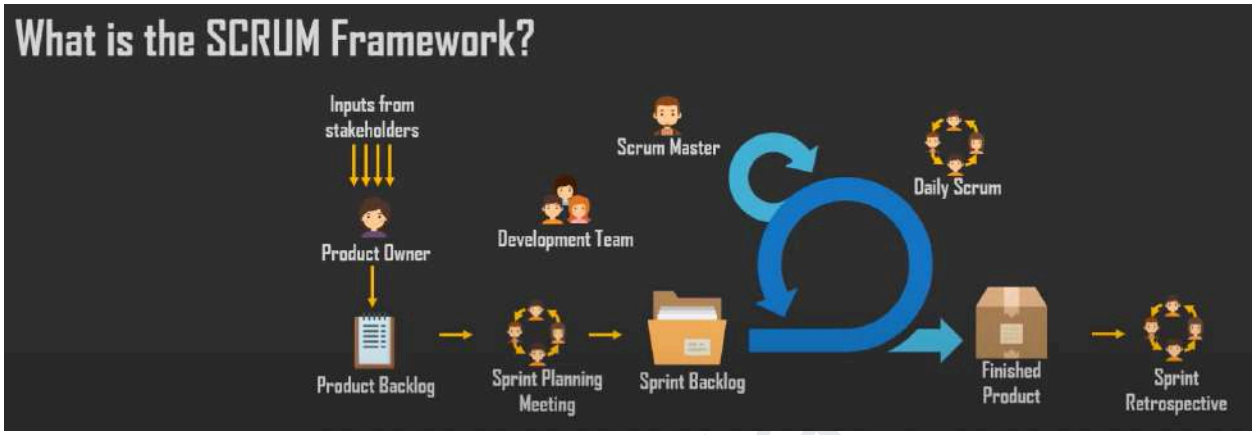
Scrum is simple to understand.

Scrum framework helps the team to work together.

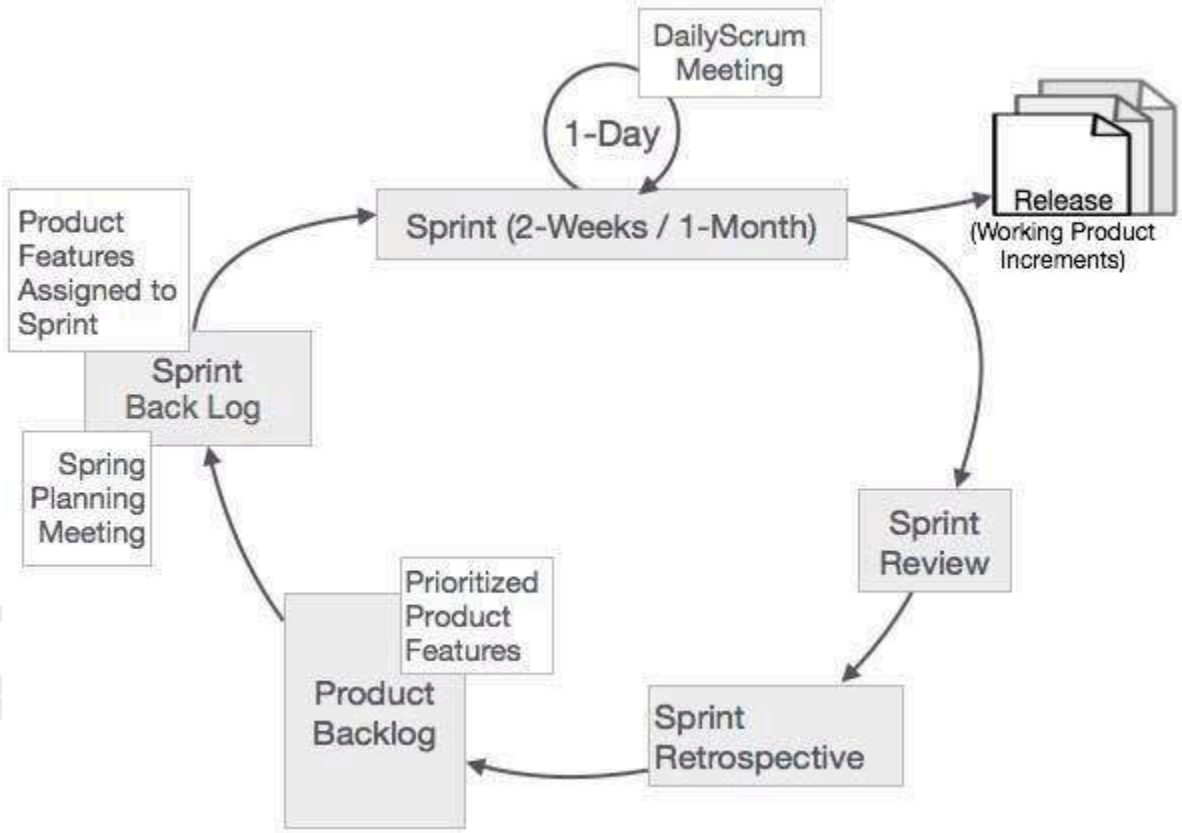



What is SCRUM?
Iteration 1
Iteration 2
Iteration 3
Iteration 4
Iteration 5
These iterations are called Sprints and at the end of each sprint you have the launch of a potentially deliverable software.
Exit full screen (f)


Product Backlog is ordered list of tasks and requirements the final product needs
Product Backlog
Sprint Backlog
List of all items from the Product Backlog that need to be worked on during a Sprint.
A burndown chart is a graphical representation of the amount of estimated remaining work.
Burndown Charts
Product Increments
Sum of work completed in a sprint, combined with all work completed during previous sprints



Scrum Process Framework



Sprint:

It breaks down big complex projects into bite-size pieces.

It makes projects more manageable, allows teams to ship high quality, work faster, and more frequently.

The sprints give them more flexibility to adapt to the changes.

Sprints are a short, time-boxed period for the Scrum team that works to complete a set amount of work.

Sprints are the core component of Scrum and agile methodology.

The right sprints will help our agile team to ship better software.

A Sprint is a time box of one month or less. A new Sprint starts immediately after the completion of the previous Sprint.

Release: When the product is completed, it goes to the Release stage.

Sprint Review: If the product still has some non-achievable features, it will be checked in this stage and then passed to the Sprint Retrospective stage.

It is held at the end of the Sprint to inspect the Increment and make changes to the Product Backlog, if needed.

Sprint Retrospective: In this stage the quality or status of the product is checked.

It occurs after the Sprint Review and prior to the next Sprint Planning. In this meeting, the Scrum Team is to inspect itself and create a plan for improvements to be enacted during the subsequent Sprint.

Product backlog: In scrum features of the product are written from the perspective of the end-user. Features are known as user stories. The collection of user stories is called a Product backlog.

It is the wish list of features of the product. After creating a product backlog.

According to the prioritized features the product is organized.

Sprint Backlog: Sprint Backlog is divided into two parts

Product features assigned to sprint

Sprint planning meeting.