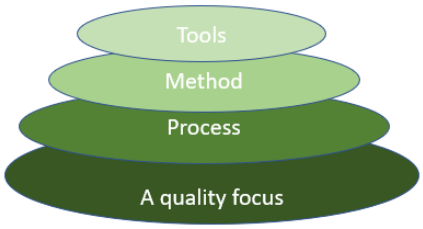
OOSE MOD-1

1.1)

➢ Layered Technology



- **A quality focus:**
  - It defines the continuous process improvement principles of software.
  - It provides integrity that means providing security to the software so that data can be accessed by only an authorized person, no outsider can access the data.
  - It also focuses on maintainability and usability.
- **Process:**
  - It is the foundation or base layer of software engineering.
  - It is key that binds all the layers together which enables the development of software before the deadline or on time.
  - The software process covers all the activities, actions, and tasks required to be carried out for software development.

    Process activities are listed below:-

    - **Communication**
    - **Planning**
    - **Modeling**
    - **Construction**
    - **Deployment**
- **Method:** During the process of software development the answers to all "how-to-do" questions are given by method. It has the information of all the tasks which includes communication, requirement analysis, design modeling, program construction, testing, and support.
- **Tools:** Software engineering tools provide a self-operating system for processes and methods. Tools are integrated which means information created by one tool can be used by another.

➢ Process Framework

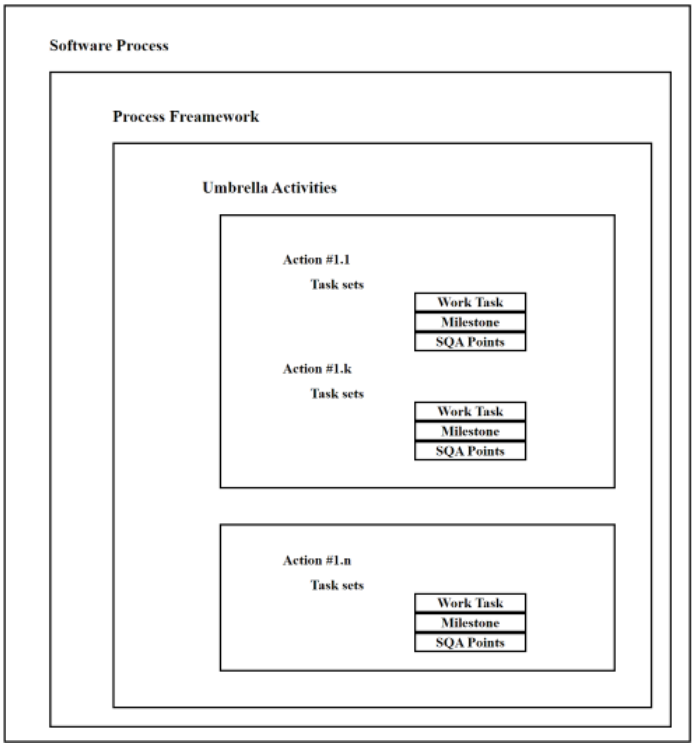
Software Process Framework details the steps and chronological order of a process.

Software Process includes:

Tasks: They focus on a small, specific objective.

Action: It is a set of tasks that produce a major work product.

Activities: Activities are groups of related tasks and actions for a major objective.

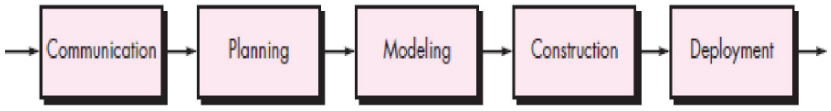


A generic process framework for software engineering defines five framework activities—communication, planning, modelling, construction, and deployment.

- **Communication**
  - By communication, customer requirement gathering is done.
  - Communication with consumers and stakeholders to determine the system's objectives and the software's requirements.
- **Planning**
  - Establish engineering work plan, describes technical risk, lists resources requirements, work produced and defines work schedule.

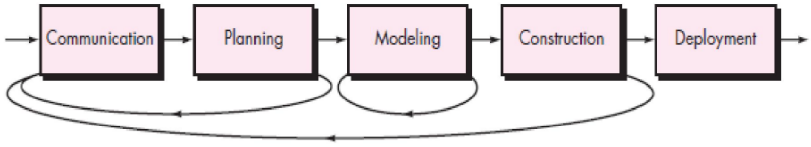
- **Modeling**
  - Architectural models and design to better understand the problem and to work towards the best solution.
  - The software model is prepared by:
    - ***Analysis of requirements***
    - ***Design***
- **Construction**
  - Creating code, testing the system, fixing bugs, and confirming that all criteria are met.
  - The software design is mapped into a code by:
    - ***Code generation***
    - ***Testing***
- **Deployment**
  - In this activity, a complete or non-complete product or software is represented to the customers to evaluate and give feedback.
  - On the basis of their feedback, we modify the product for the supply of better products.

process flow—describes how the framework activities and the actions and tasks that occur within each framework activity are organized with respect to sequence and time.
Different types of process flow are:
– Linear



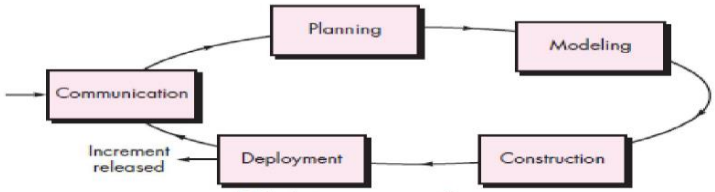
(a) Linear process flow

– Iterative



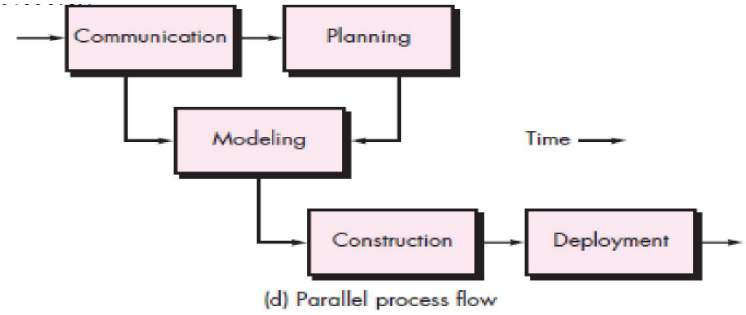
(b) Iterative process flow

– evolutionary



(c) Evolutionary process flow

– Parallel



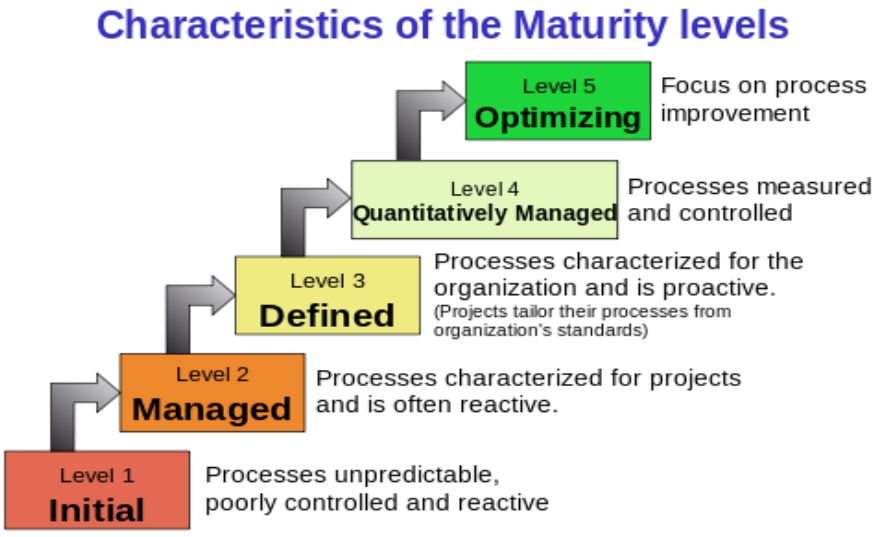
(d) Parallel process flow

➢ Capability Maturity Model(CMMI)

- CMMI stands for Capability Maturity Model Integration.
- It is a process improvement framework that helps organizations improve their processes and ultimately their performance.
- CMMI provides a set of best practices for developing, managing, and maintaining products and services.

Objectives of CMMI :

- Fulfilling customer needs and expectations.
- Value creation for investors/stockholders.
- Market growth is increased.
- Improved quality of products and services.
- Enhanced reputation in Industry.



CMMI Model – Maturity Levels : In CMMI with staged representation, there are five maturity levels described as follows :

- **Maturity level 1 : Initial**
  - processes are poorly managed or controlled.
  - unpredictable outcomes of processes involved.
  - ad hoc and chaotic approach used.
  - Lowest quality and highest risk.
- **Maturity level 2 : Managed**
  - processes are planned and controlled.
  - projects are managed and implemented according to their documented plans.
  - This risk involved is lower than Initial level, but still exists.
  - Quality is better than Initial level.
- **Maturity level 3 : Defined**
  - processes are well characterized and described using standards, proper procedures, and methods, tools, etc.
  - Medium quality and medium risk involved.
  - Focus is process standardization.
- **Maturity level 4 : Quantitatively managed**
  - quantitative objectives for process performance and quality are set.
  - higher quality of processes is achieved.
  - lower risk
- **Maturity level 5 : Optimizing**
  - continuous improvement in processes and their performance.
  - improvement has to be both incremental and innovative.
  - highest quality of processes.
  - lowest risk in processes and their performance.

1.2)

➢ Prescriptive Process Model

Prescriptive process models act as roadmaps for completing complex projects, particularly in software development. They define a clear set of activities, deliverables, and milestones that need to be accomplished in a specific order.

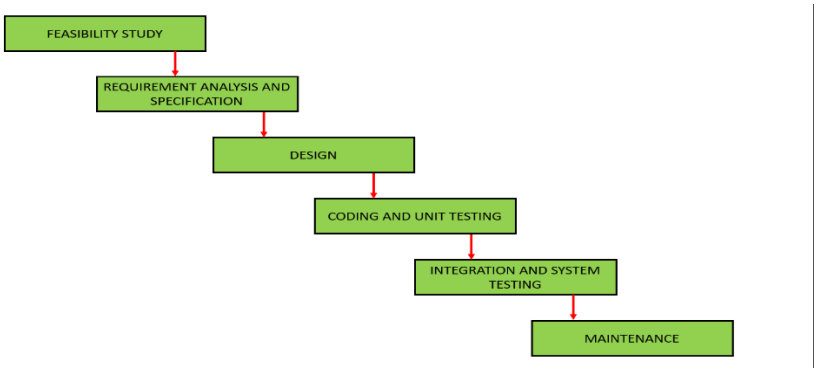
Key Characteristics:

- ***Structured:*** Activities follow a defined sequence, often with dependencies between stages.

- ***Phased:*** The project is divided into distinct phases, each with clear deliverables and completion criteria.
- ***Documented:*** The process is documented and communicated to all stakeholders.
- ***Controlled:*** Change management processes are in place to manage deviations from the plan.

1. Waterfall Model

The waterfall model is useful in situations where the project requirements are well-defined and the project goals are clear. It is often used for large-scale projects with long timelines, where there is little room for error.



i. **Feasibility Study:** The main goal of this phase is to determine whether it would be financially and technically feasible to develop the software.
ii. **Requirements analysis and specification phase:** The aim of this phase is to understand the exact requirements of the customer and to document them properly. Both the customer and the software developer work together so as to document all the functions, performance, and interfacing requirement of the software.
iii. **Design Phase:** This phase aims to transform the requirements gathered in the SRS into a suitable form which permits further coding in a programming language. It defines the overall software architecture together with high level and detailed design.
iv. **Coding & Unit Testing:** In the coding phase software design is translated into source code using any suitable programming language. Thus each designed module is coded. The aim of the unit testing phase is to check whether each module is working properly or not.
v. **Integration and System testing:** Integration of different modules is undertaken soon after they have been coded and unit tested. Integration of various modules is carried out incrementally over a number of steps.
vi. **Maintenance:** Maintenance is the most important phase of a software life cycle. The effort spent on maintenance is 60% of the total effort spent to develop a full software.

Advantages of Classical Waterfall Model

- ***Easy to Understand:*** Classical Waterfall Model is very simple and easy to understand.
- ***Individual Processing:*** Phases in the Classical Waterfall model are processed one at a time.
- ***Properly Defined:*** In the classical waterfall model, each stage in the model is clearly defined.
- ***Clear Milestones:*** Classical Waterfall model has very clear and well-understood milestones.
- ***Properly Documented:*** Processes, actions, and results are very well documented.
- ***Reinforces Good Habits:*** Classical Waterfall Model reinforces good habits like define-before-design and design-before-code.
- ***Working:*** Classical Waterfall Model works well for smaller projects and projects where requirements are well understood.

Disadvantages of Classical Waterfall Model

- **No Feedback Path:**
  In the classical waterfall model evolution of software from one phase to another phase is like a waterfall. It assumes that no error is ever committed by developers during any phase. Therefore, it does not incorporate any mechanism for error correction.
- **Difficult to accommodate Change Requests:**
  It is difficult to accommodate any change requests after the requirements specification phase is complete.
- **Limited Flexibility:**
  The Waterfall Model is a rigid and linear approach to software development, which means that it is not well-suited for projects with changing or uncertain requirements.
- **Not Suitable for Complex Projects:**
  The Waterfall Model is not well-suited for complex projects, as the linear and sequential nature of the model can make it difficult to manage multiple dependencies and interrelated components.

Applications of Classical Waterfall Model

- *Safety-Critical Systems*
- *Government and Defense Projects*

2. Incremental Model

- Incremental Model is a process of software development where requirements divided into multiple standalone modules of the software development cycle.

- In this model, each module goes through the requirements, design, implementation and testing phases.

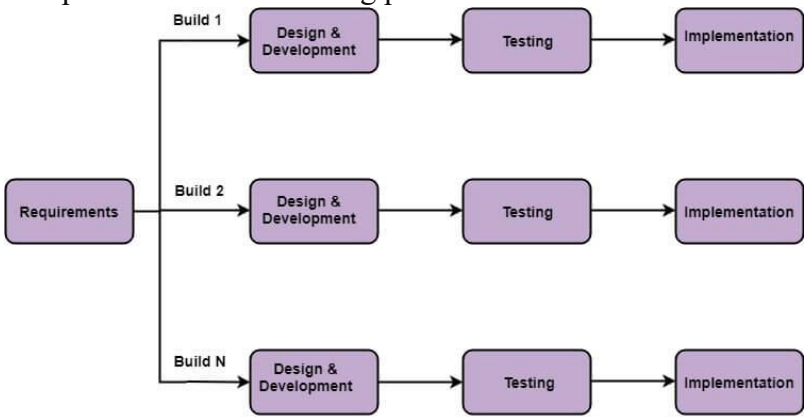


Fig: Incremental Model

i. **Requirement analysis:** In the first phase of the incremental model, the product analysis expertise identifies the requirements. To develop the software under the incremental model, this phase performs a crucial role.
ii. **Design & Development:** In this phase of the Incremental model of SDLC, the design of the system functionality and the development method are finished with success.
iii. **Testing:** In the testing phase, the various methods are used to test the behavior of each task.
iv. **Implementation:** Implementation phase enables the coding phase of the development system. It involves the final coding that design in the designing and development phase and tests the functionality in the testing phase. After completion of this phase, the number of the product working is enhanced and upgraded up to the final system product

Advantages of Incremental Model

- Errors are easy to be recognized.
- Easier to test and debug
- More flexible.
- Simple to manage risk because it handled during its iteration.
- The Client gets important functionality early.

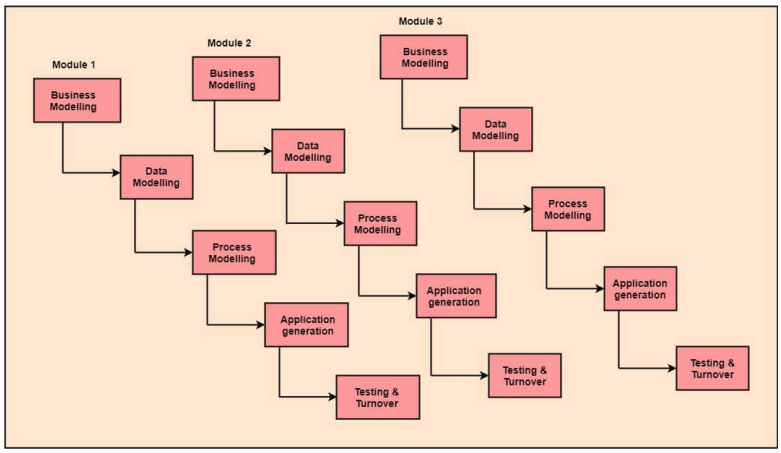
Disadvantages of Classical Incremental Model

- Need for good planning
- Total Cost is high.
- Well defined module interfaces are needed.

3. RAD Model

- The rapid application development model is type of incremental software process model in which there is extremely short development cycle.
- This model is similar to waterfall model which achieves the high speed development using component based construction.

Fig: RAD Model



i. **Business Modelling:** The information flow among business functions is defined by answering questions like what data drives the business process, what data is generated, who generates it, where does the information go, who process it and so on.
ii. **Data Modelling:** The data collected from business modeling is refined into a set of data objects (entities) that are needed to support the business.
iii. **Process Modelling:** The information object defined in the data modeling phase are transformed to achieve the data flow necessary to implement a business function.
iv. **Application Generation:** Automated tools are used to facilitate construction of the software; even they use the 4th GL techniques.
v. **Testing & Turnover:** Many of the programming components have already been tested since RAD emphasis reuse. This reduces the overall testing time. But the new part must be tested, and all interfaces must be fully exercised.

When to use RAD Model?

- When the requirements are well-known.
- When the technical risk is limited.
- When there's a necessity to make a system, which modularized in 2-3 months of period.
- It should be used only if the budget allows the use of automatic code generating tools.

Advantages of RAD Model

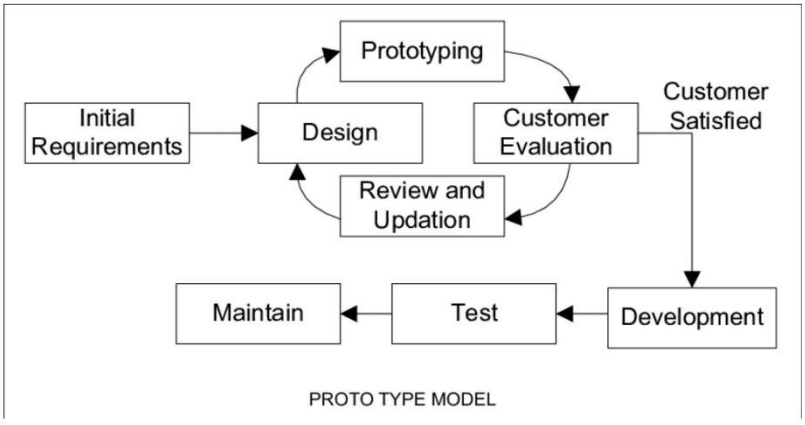
- This model is flexible for change.
- In this model, changes are adoptable.
- Each phase in RAD brings highest priority functionality to the customer.
- It reduced development time.
- It increases the reusability of features.

Disadvantages of RAD Model

- It required highly skilled designers.
- All application is not compatible with RAD.
- For smaller projects, we cannot use the RAD model.
- On the high technical risk, it's not suitable.
- Required user involvement.

4. Prototyping Model

- 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.



PROTO TYPE MODEL

Advantages of Prototype Model

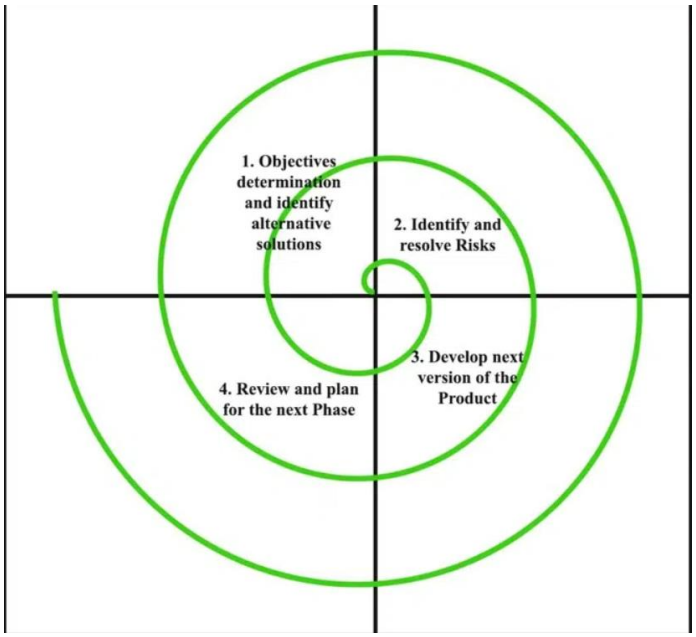
- Reduce the risk of incorrect user requirement
- Good where requirement are changing/uncommitted
- Regular visible process aids management
- Support early product marketing
- Reduce Maintenance cost.
- Errors can be detected much earlier as the system is made side by side.

Disadvantages of Prototype Model

- An unstable/badly implemented prototype often becomes the final product.
- Require extensive customer collaboration
    - ✓ Costs customer money
    - ✓ Needs committed customer
    - ✓ Difficult to finish if customer withdraw
    - ✓ May be too customer specific, no broad market
- Difficult to know how long the project will last.

5. Spiral Model

- The Spiral Model is a Software Development Life Cycle (SDLC) model that 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. Each loop of the spiral is called a Phase of the software development process.



i. **Planning:** The first phase of the Spiral Model is the planning phase, where the scope of the project is determined and a plan is created for the next iteration of the spiral.
ii. **Risk Analysis:** In the risk analysis phase, the risks associated with the project are identified and evaluated.
iii. **Engineering**: In the engineering phase, the software is developed based on the requirements gathered in the previous iteration.
iv. **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.
v. **Planning**: The next iteration of the spiral begins with a new planning phase, based on the results of the evaluation.

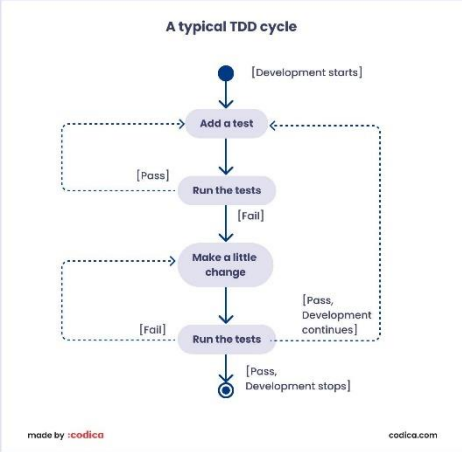
Advantages of the Spiral Model

- **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.

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.
- **Difficulty in time management:** As the number of phases is unknown at the start of the project, time estimation is very difficult.
- **Time-Consuming:** The Spiral Model can be time-consuming, as it requires multiple evaluations and reviews.

6. Test Driven Development



Advantages of TDD

- **Improved Code Quality:** TDD encourages developers to write modular, clean, and maintainable code since they need to pass tests continuously.
- **Early Bug Detection:** Bugs are caught early in the development cycle, making them cheaper and easier to fix.
- **Rapid Feedback:** Developers receive immediate feedback on their code changes, helping them identify and rectify issues quickly.
- **Refactoring Confidence:** Developers can refactor code confidently, knowing that if they inadvertently break something, the tests will catch it.

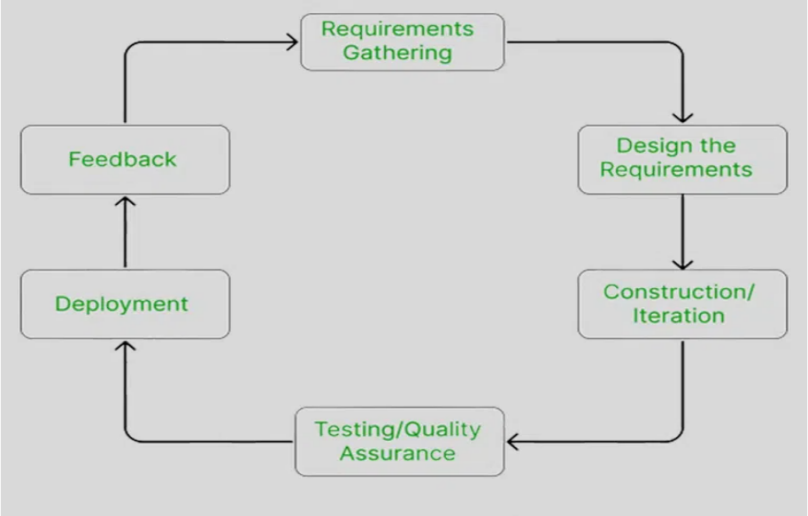
Disadvantages of TDD

- **Learning Curve:** TDD requires a mindset shift and discipline. Developers need to learn how to write effective tests and adhere to the TDD workflow, which can be challenging initially.
- **Test Maintenance Overhead:** Maintaining a comprehensive test suite can sometimes be time-consuming and may require significant effort, especially as the codebase grows.

1.3)

➢ Agile Process

- "**Agile process model**" refers to a software development approach based on iterative development.
- Agile methods break tasks into smaller iterations, or parts do not directly involve long term planning.
- The project scope and requirements are laid down at the beginning of the development process.



i. **Requirements gathering:** In this phase, you must define the requirements. You should explain business opportunities and plan the time and effort needed to build the project.
ii. **Design the requirements:** When you have identified the project, work with stakeholders to define requirements. You can use the user flow diagram or the high-level UML diagram to show the work of new features and show how it will apply to your existing system.
iii. **Construction/ iteration:** When the team defines the requirements, the work begins. The product will undergo various stages of improvement, so it includes simple, minimal functionality.
iv. **Testing:** In this phase, the Quality Assurance team examines the product's performance and looks for the bug.
v. **Deployment:** In this phase, the team issues a product for the user's work environment.
vi. **Feedback:** After releasing the product, the last step is feedback. In this, the team receives feedback about the product and works through the feedback.

Advantages of Agile Process

- Frequent Delivery.
- Face-to-face Communication with clients.
- Efficient design and fulfils the business requirement.
- Anytime changes are acceptable.
- It reduces total development time.

Disadvantages of Agile process

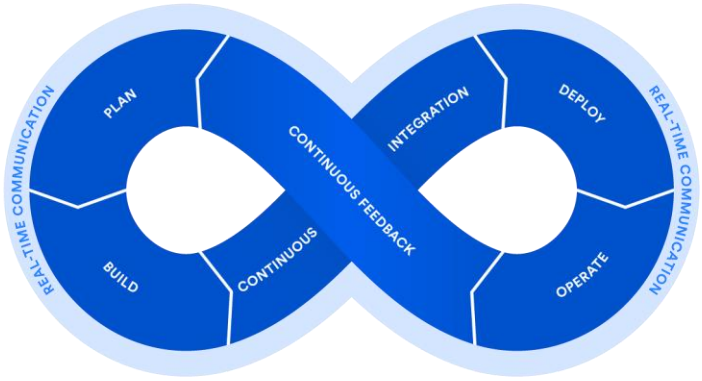
- Due to the shortage of formal documents, it creates confusion and crucial decisions taken throughout various phases can be misinterpreted at any time by different team members.
- Due to the lack of proper documentation, once the project completes and the developers allotted to another project, maintenance of the finished project can become a difficulty.

➢ Scrum Industry Perspective

From an industry perspective, Scrum is widely regarded as one of the most popular and effective Agile frameworks for managing software development projects. Here's how Scrum is viewed within the industry:

- **Flexibility and Adaptability:**
  - Scrum is highly valued for its flexibility and adaptability to changing requirements.
  - In industries where requirements are prone to change or where innovation is key, Scrum allows teams to respond quickly to evolving needs and priorities.
- **Iterative and Incremental Approach**:
  - The iterative and incremental nature of Scrum enables teams to deliver working software in short iterations, typically lasting 2-4 weeks.
  - This approach allows for early and frequent feedback from stakeholders, reducing the risk of project failure and ensuring that the delivered product meets customer expectations.
- **Cross-functional Collaboration:**
  - Scrum promotes collaboration and self-organization among cross-functional teams, including developers, testers, designers, and product owners.
  - This collaborative approach fosters a sense of ownership and accountability within the team, leading to higher levels of productivity and morale.

➢ Devops Development Practice



- DevOps is a set of practices, principles, and cultural philosophies that aim to improve collaboration and communication between software development (Dev) and IT operations (Ops) teams.
- It focuses on automating processes, adopting agile methodologies, and fostering a culture of continuous integration, continuous delivery (CI/CD), and continuous improvement.
    - **Automation:** DevOps promotes automation to reduce manual effort, minimize errors, and increase efficiency in tasks like build, testing, and deployment processes.
    - **Continuous Integration (CI):** Integrating code changes into a shared repository frequently triggers automated builds and tests to detect and address integration errors early in the development process.
    - **Continuous Delivery (CD):** Automating the deployment process ensures software can be deployed to production environments quickly and reliably at any time, leveraging practices like automated deployment pipelines and infrastructure as code (IaC).
    - **Collaboration and Communication:** DevOps fosters collaboration and communication across development, operations, and stakeholders to break down silos and promote shared responsibility and accountability.
    - **Lean and Agile Practices:** DevOps incorporates Lean and Agile principles to reduce waste, improve flow, and embrace iterative and incremental development practices.
    - **Feedback Loops:** DevOps creates feedback loops throughout the software delivery lifecycle to gather insights, identify bottlenecks, and drive continuous improvement.