Unit 1 Software Development Process Models

L–6+T–3 Hours

- Introduction to Software Engineering - Software Development process models – Waterfall Model, Incremental Model, Spiral Model, V Model – Agile Development- Agile Methodologies- Agile Principles- SCRUM and XP - Project management- Stages of Project management- 4 P's of project management - Process & Project metrics. Case Study: The Railways tracking - Arrival Time Prediction System

Software Engineering

- Understand the problem before Software solution development
- Design must a pivotal activity
- Software should exhibit high quality
- Software should be maintainable
- Software in all of its forms and across all of its application domains should be engineered

The IEEE [IEE93a] has developed a more comprehensive definition when it states:

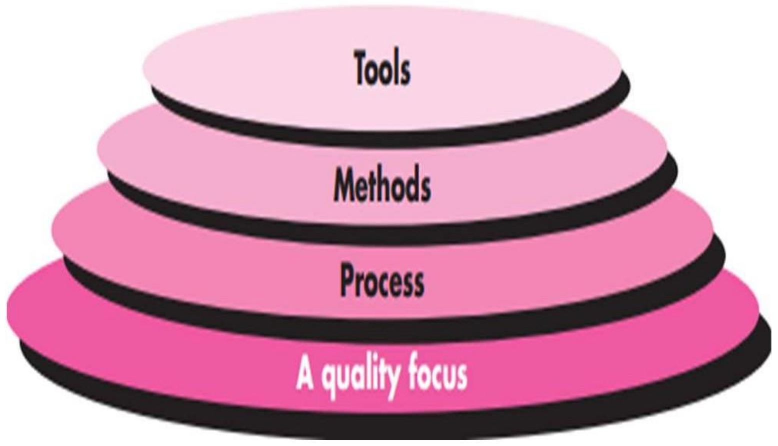
(1)The application of a systematic, disciplined, quantifiable approach.to the development, operation, and maintenance of software; that is, the application of engineering to software.

(2)The study of approaches as in (1)

Software Engineering Layers

- ❑ Process – Framework for effective SE technology delivery
  - ✓ Basis for management control of software projects
  - ✓ Establishes the context to
    - ➢ Apply Technical products
    - ➢ Produce work products
    - ➢ Achieve mile stones
    - ➢ Ensure quality
    - ➢ Manage the changes
- ❑ Methods – Technical how-to's for building software
  - ✓ Includes communication, requirements analysis, design modeling, program construction, testing, and support
  - ✓ Includes modeling activities and other descriptive techniques
- ❑ Tools – Automated or semiautomated support for the process and the methods
  - ✓ Tools are integrated to establish Computer-Aided Software Engineering

LAYERED TECHNOLOGY

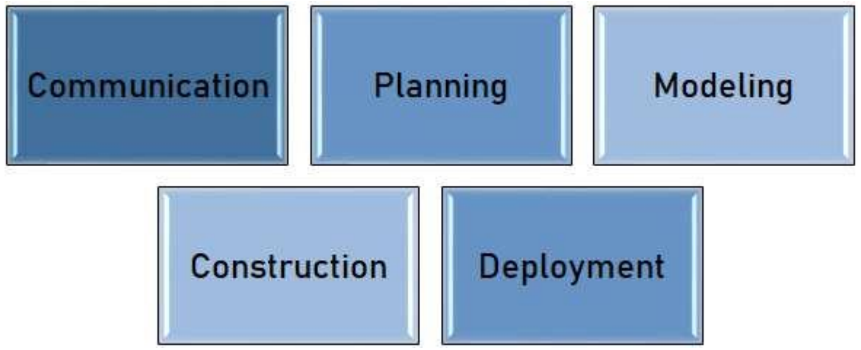


The Software Process

- Process - Collection of activities, actions, and tasks to create a work product
- Activity strives to achieve a broad objective
  - Communication with stakeholders
  - Regardless of
    - Application domain,
    - Size of the project,
    - Complexity of the effort,
    - Degree of rigor with which software engineering is to be applied
- Action – Set of tasks that produce a major work product (e.g., an architectural design model)
  - An architectural design
- Task – focuses on a small, but well-defined objective (e.g., conducting an Unit test)
  - Produces a tangible outcome

The Software Process – Process framework

- Foundation for a complete SE process by identifying a small number of framework activities
  - ✓ Activities applicable to all software projects, regardless of their size or complexity
  - ✓ Encompasses umbrella activities applicable across the entire software process



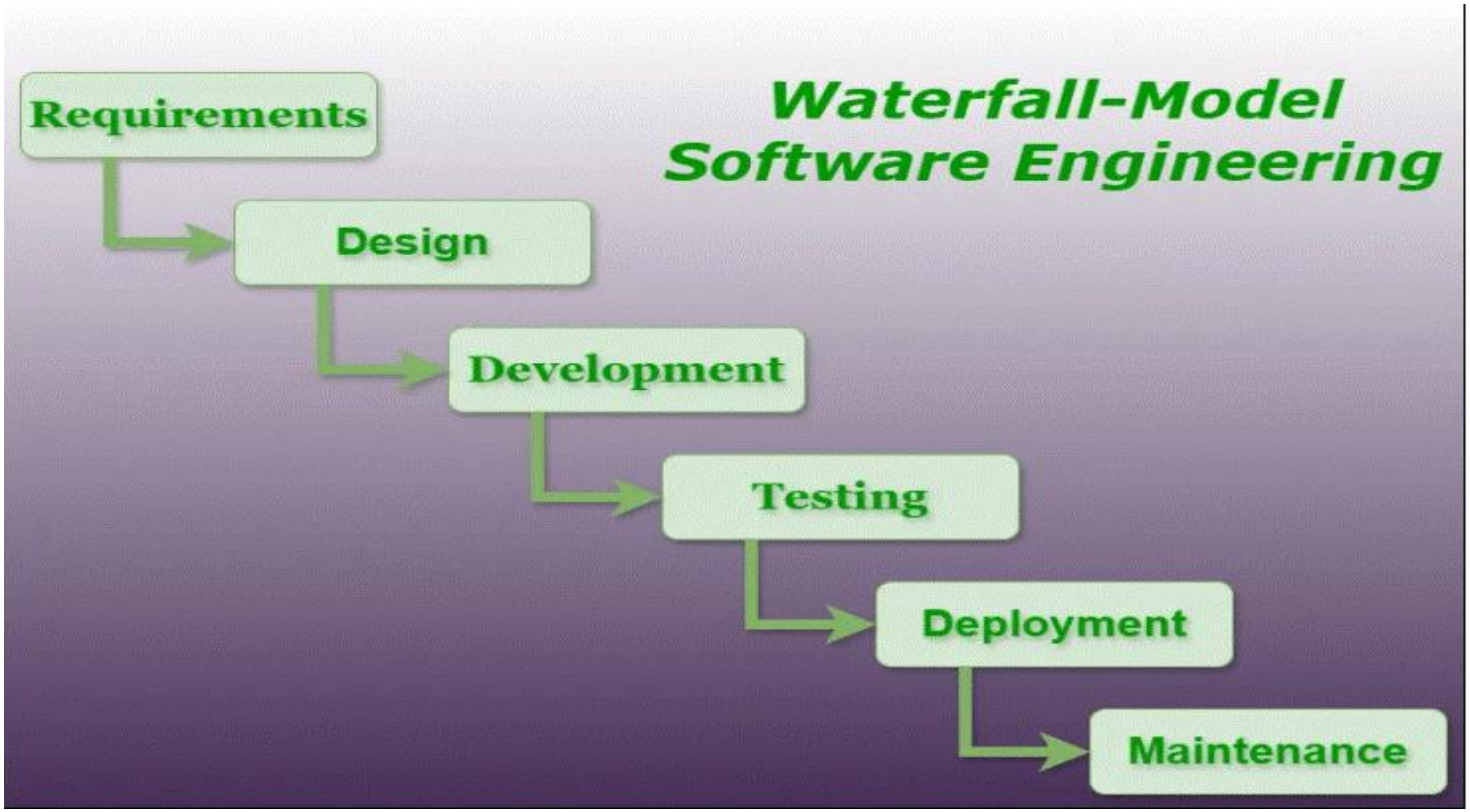
Process Framework Activities

- Framework activities can be used during the development of
  - ✓ Small and simple programs,
  - ✓ Large Web applications,
  - ✓ Large, complex computer-based system

WATERFALL MODEL

- The **Waterfall Model** is a **Traditional Software Development Methodology**. It was first introduced by **Winston W. Royce**
- It is a linear and sequential approach to software development that consists of several phases.
- The waterfall model is a Software Development Model used in the context of large, complex projects, typically in the field of information technology. It is characterized by a structured, sequential approach to **Project Management** and **Software Development**.

- **Phases of Waterfall Model**
- **Classical Waterfall Model** divides the life cycle into a set of phases. The development process can be considered as a sequential flow in the waterfall. The different sequential phases of the classical waterfall model are follow:


Waterfall-Model
Software Engineering
Requirements
Design
Development
Testing
Deployment
Maintenance

- **1. Requirements Analysis and Specification**
- Requirement Analysis and specification phase aims to understand the exact requirements of the customer and document them properly. This phase consists of two different activities.
- **1. Requirement Gathering and Analysis:** Firstly all the requirements regarding the software are gathered from the customer and then the gathered requirements are analyzed.
- The goal of the analysis part is to remove incompleteness and inconsistencies

- **2. Requirement Specification:** These analyzed requirements are documented in a software requirement specification (SRS) document. SRS document serves as a contract between the development team and customers. Any future dispute between the customers and the developers can be settled by examining the SRS document.

- **Design**
- The goal of this **Software Design Phase** is to convert the requirements acquired in the SRS into a format that can be coded in a programming language. It includes high-level and detailed design as well as the overall software architecture. A **Software Design Document** is used to document all of this effort (SDD).

- **High-Level Design (HLD)**: This phase focuses on outlining the broad structure of the system. It highlights the key components and how they interact with each other, giving a clear overview of the system’s architecture.
- **Low-Level Design (LLD)**: Once the high-level design is in place, this phase zooms into the details. It breaks down each component into smaller parts and provides specifics about how each part will function, guiding the actual coding process.

- **3. Development**
- In the **Development Phase** software design is translated into source code using any suitable programming language. Thus each designed module is coded. The unit testing phase aims to check whether each module is working properly or not.

- In this phase, developers begin writing the actual source code based on the designs created earlier.
- The goal is to transform the design into working code using the most suitable programming languages.
- Unit tests are often performed during this phase to make sure that each component functions correctly on its own.

4. Testing and Deployment

- **1. 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 several steps.

- Alpha testing is the system testing performed by the development team.
- **Beta testing**: Beta testing is the system testing performed by a friendly set of customers.
- **Acceptance testing**: After the software has been delivered, the customer performs acceptance testing to determine whether to accept the delivered software or reject it.

5. Maintenance

- In **Maintenance Phase** 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. There are three types of maintenance.

- **Corrective Maintenance:** This type of maintenance is carried out to correct errors that were not discovered during the product development phase.
- **Perfective Maintenance:** This type of maintenance is carried out to enhance the functionalities of the system based on the customer’s request.
- **Adaptive Maintenance:** Adaptive maintenance is usually required for porting the software to work in a new environment such as working on a new computer platform or with a new operating system.

Features of Waterfall Model

- Following are the features of the waterfall model:
- **Sequential Approach**: The waterfall model involves a sequential approach to software development, where each phase of the project is completed before moving on to the next one.
- **Document-Driven:** The waterfall model depended on documentation to ensure that the project is well-defined and the project team is working towards a clear set of goals.

- **Quality Control:** The waterfall model places a high emphasis on quality control and testing at each phase of the project, to ensure that the final product meets the requirements and expectations of the stakeholders.
- **Rigorous Planning**: The waterfall model involves a careful planning process, where the project scope, timelines, and deliverables are carefully defined and monitored throughout the project lifecycle.

Advantages of Waterfall Model

- **Easy to Understand:** The 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:** The classical Waterfall model has very clear and well-understood milestones.
- **Properly Documented:** Processes, actions, and results are very well documented.
- **Reinforces Good Habits:** The 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 Waterfall Model

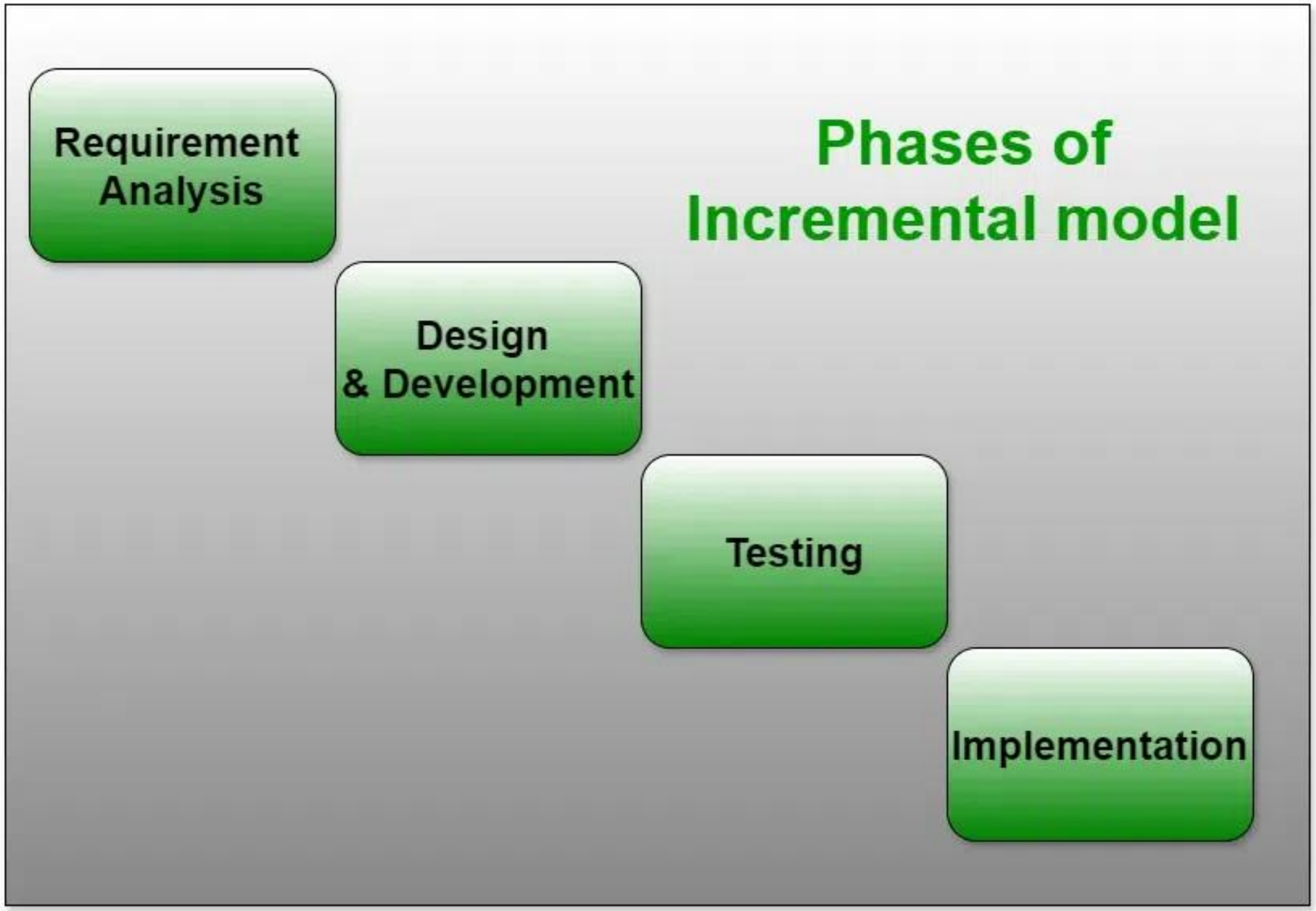
- Inflexibility to Changes
- Late Testing
- Poor for Complex & Evolving Projects
- No Working Software Until Late
- Risk of Obsolescence
- Lack of User Involvement

Incremental Process Model

- The Incremental model is a software Development approach which is used to breakdown the project into smaller and easily manageable parts.
- In these, each part passes through Requirement, Design, Testing phases and Implementation phase. The overall process continue until we got the complete System.

Key Characteristics of Incremental Process Model

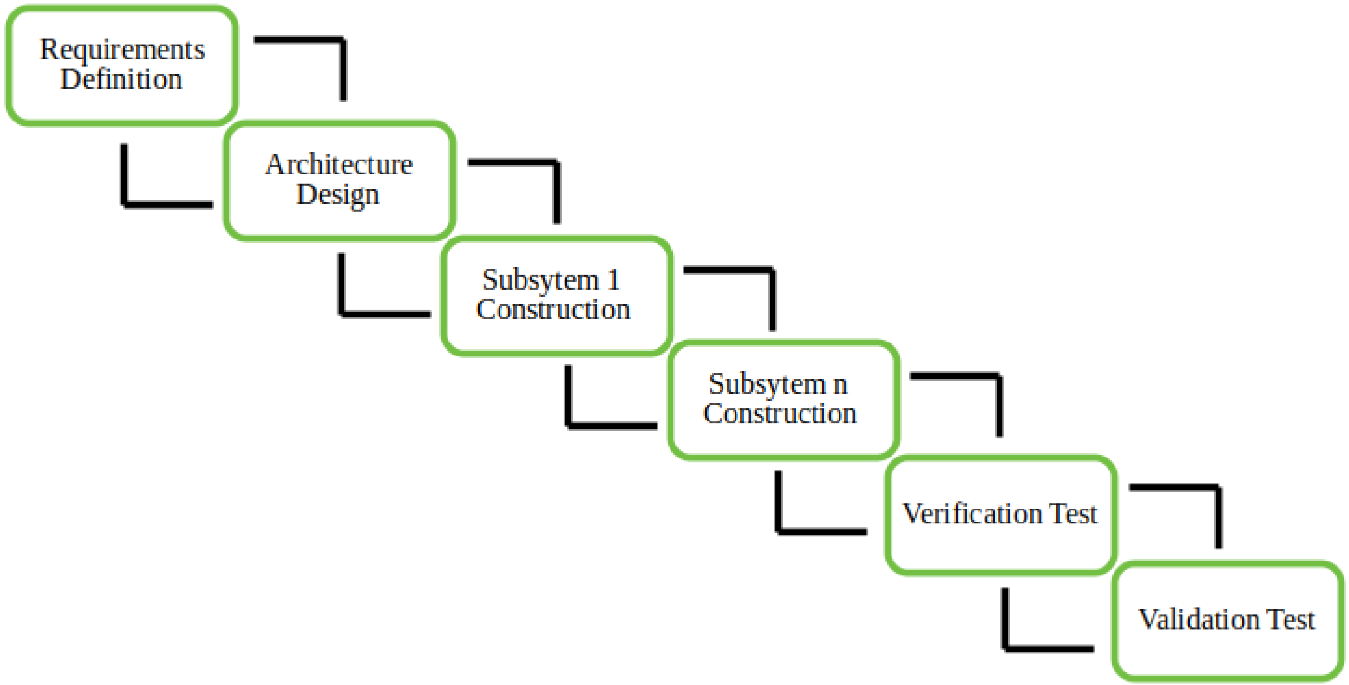
- **Partial System Delivery:** The system is developed and delivered in small, manageable pieces. Each part adds new features to the previous version.
- **Early Functionality:** Basic functionality is available early in the project. This allows users to start using and testing the system quickly.
- **Customer Feedback Loop:** Feedback is collected after each part is delivered. This helps improve the next version of the system.
- **Flexible to Changes:** Changes or new features can be added between increments. This makes the model flexible to evolving needs.
- **Combination of Linear and Iterative Approaches:** Combines the structured approach of Waterfall with flexibility. Supports both planning and ongoing improvements.


Phases of Incremental model
Requirement Analysis
Design & Development
Testing
Implementation

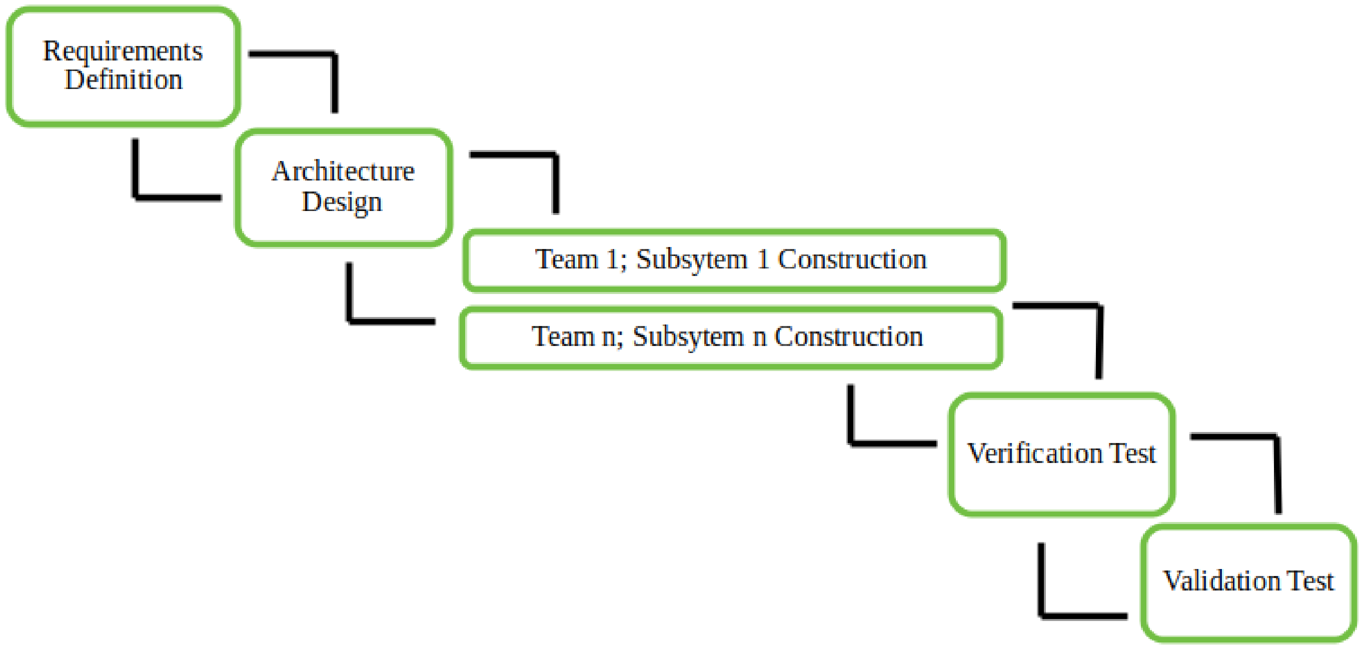
Types of Incremental Model

- **Staged Delivery Model**
- **Parallel Development Model**

Staged Delivery Model



Parallel Development Model



Use Cases of Incremental Process Model

- When the requirements are well-defined and clear
- If the project has a long development timeline
- If the customer needs a quick product release
- When you want to develop the most important features first

Advantages of Incremental Process Model

- Faster Software Delivery
- Clear Understanding for Clients
- Easy to Implement Changes
- Effective Risk Management
- Flexible Criteria and Scope
- Cost-Effective
- Simpler Error Identification

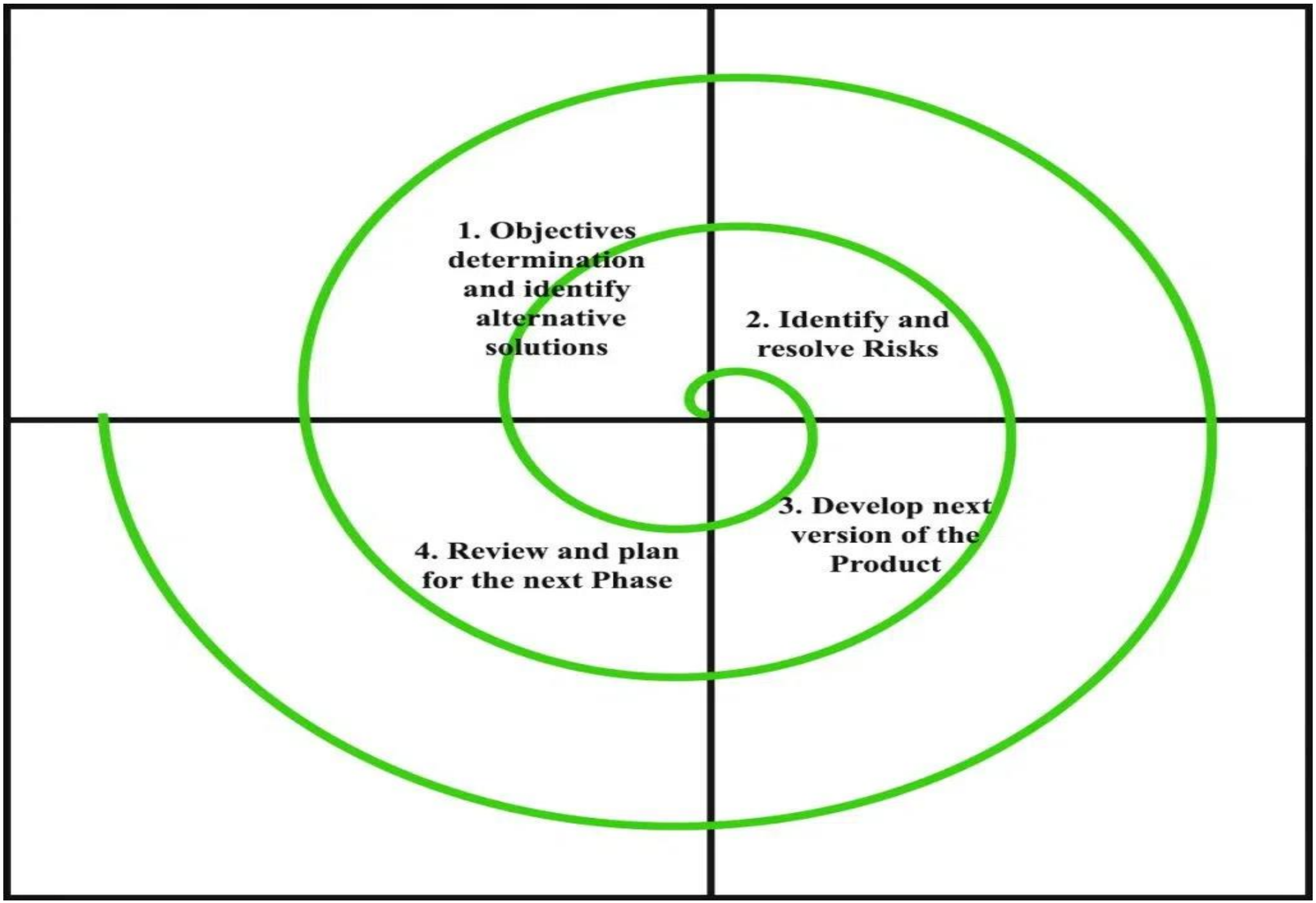
Disadvantages of Incremental Process Model

- **Requires a Skilled Team and Proper Planning**
- **Cost Can Increase Over Time**
- **Incomplete Requirement Gathering Can Cause Design Issues**
- **Lack of Smooth Flow Between Increments**
- **High Effort to Fix Repeated Issues**

Spiral Model

- **The Spiral Model** is one of the most important SDLC model. The Spiral Model is a combination of the waterfall model and the iterative model. It provides support for **Risk Handling**.
- The Spiral Model was first proposed by **Barry Boehm**.

- 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, the project manager has an important role in developing a product using the spiral model.
- 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.


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

- **1. Objectives Defined**
- functional and non-functional requirements.
- Requirements are gathered from the customers and the objectives are identified, elaborated, and analyzed at the start of every phase.

- **2. Risk Analysis and Resolving**
- In the risk analysis phase, the risks associated with the project are identified and evaluated.
- 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

- **3. Develop the 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.
- 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.

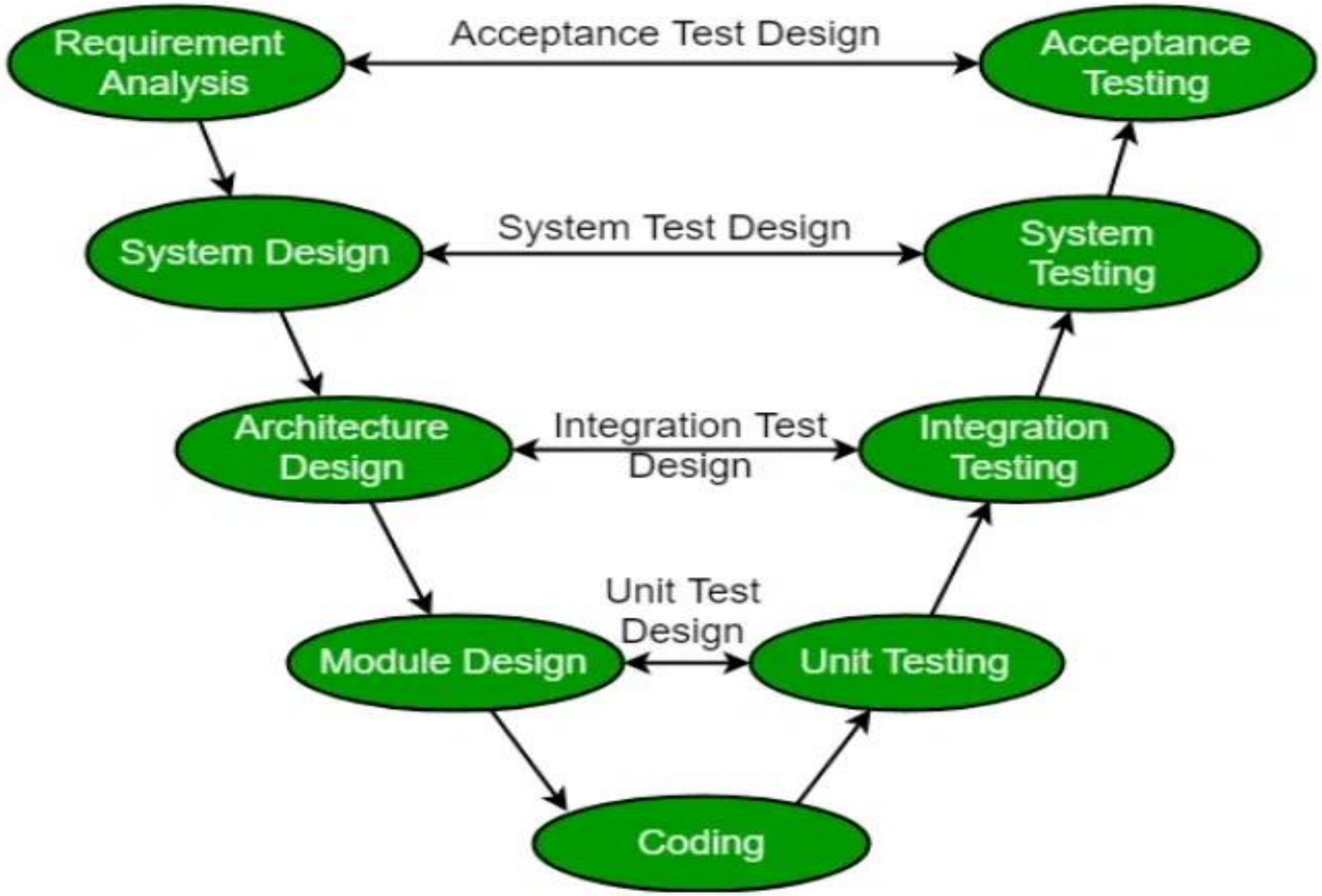
V MODEL

- The **SDLC V-Model** is a Types of **Software Development Life Cycle (SDLC),** which is used in Software Development process. In V-Model is the extension of the Traditional Software Development Model.

- It is creating a Structure like the "V" which includes the different phases which we are discussing here in detailed manner.

Phases of SDLC V-Model

- The V-Model, which includes the **Verification and Validation** it is a structural approach to the software development. The following is the different **Phases of V-Model** of the SDLC.


Requirement Analysis
Acceptance Test Design
Acceptance Testing
System Design
System Test Design
System Testing
Architecture Design
Integration Test Design
Integration Testing
Unit Test Design
Module Design
Unit Testing
Coding

1. Business Requirement Analysis

- This is the first step of the designation of the development cycle where product requirement needs to be cured from the customer's perspective. in these phases include proper communication with the customer to understand the requirements of the customers.

2. System Design

- In this phase, the overall structure of the software is planned out. The team develops both the high-level design (how the system will be structured) and detailed design (how the individual components will work).

3. Architectural Design

- In this stage, architectural specifications are comprehended and designed. Usually, several technical approaches are put out, and the ultimate choice is made after considering both the technical and financial viability

4. Module Design

- This phase, known as Low-Level Design (LLD), specifies the comprehensive internal design for every system module. Compatibility between the design and other external systems as well as other modules in the system architecture is crucial

5. Coding Phase

- This is where the software is actually built. Developers write the code based on the design created in the previous phase.
- The Coding step involves writing the code for the system modules that were created during the Design phase.

V-Model Validation Phases

- It involves dynamic analysis techniques (functional, and non-functional), and testing done by executing code. Validation is the process of evaluating the software after the completion of the development phase to determine whether the software meets the customer's expectations and requirements.

Importance of V-Model

- 1. Early Defect Identification
- 2. Determining the Phases of Development and Testing
- 3. Prevents "Big Bang" Testing
- 4. Improves Cooperation
- 5. Improved Quality Assurance

Advantages of V-Model

- This is a highly disciplined model and Phases are completed one at a time.
- V-Model is used for small projects where project requirements are clear.
- Simple and easy to understand and use.
- This model focuses on verification and validation activities early in the life cycle thereby enhancing the probability of building an error-free and good quality product.
- It enables project management to track progress accurately.

Dis-Advantages of V-Model

- The V-Model is a linear and sequential model, which can make it difficult to adapt to changing requirements or unexpected events.
- The V-Model can be time-consuming, as it requires a lot of documentation and testing.
- High risk and uncertainty.

Agile Development Models

- 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 customer change requests during project development and the high cost and time required to incorporate these changes

Steps in the Agile Model

- Requirement Gathering
- Design the Requirements
- Construction / Iteration
- Testing / Quality Assurance
- Deployment
- Feedback

Agile SDLC Methods

- **Test-Driven Development (TDD)**
- **Behavior Driven Development (BDD)**
- **Exploratory Testing**
- **Extreme Programming (XP)**

- Extreme programming is a customer-oriented methodology that helps to deliver a good quality product that meets customer expectations and requirements.

Principles of the Agile Model

- Our highest priority is to satisfy the client through early and continuous delivery of valuable computer software.
- Welcome dynamic necessities, even late in development. **Agile Processes** harness modification for the customer’s competitive advantage.
- Deliver operating computer software often, from a pair of weeks to a couple of months, with a preference to the shorter timescale.
- Business individuals and developers should work along daily throughout the project.

- The build comes around actuated people. offer them the setting and support they have, and trust them to urge the task done.
- the foremost economical and effective methodology of conveyancing info to and among a development team is face-to-face speech.
- Working with computer software is the primary life of progress.
- Agile processes promote property development. The sponsors, developers, and users will be able to maintain a relentless pace indefinitely.

- Continuous attention to technical excellence and smart style enhances nimbleness.
- Simplicity the art of maximizing the number of work not done is essential.
- the most effective architectures, necessities, and styles emerge from self–organizing groups.
- At regular intervals, the team reflects on a way to become simpler, then tunes and adjusts its behavior consequently.

When To Use the Agile Model?

- When new changes need to be implemented. The freedom that playfulness gives for change is very important.
- New changes can be implemented at a very low cost due to the frequency of new increments.
- Implementing a new feature requires developers to lose only a few days or even just hours of work to get it back and implemented.

- In the Agile model, unlike the Waterfall model, very limited planning is required to start a project.
- Agile believes that the needs of end users are always changing in the dynamic business and IT world.
- Changes can be discussed and features can be redesigned or removed based on feedback.

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.

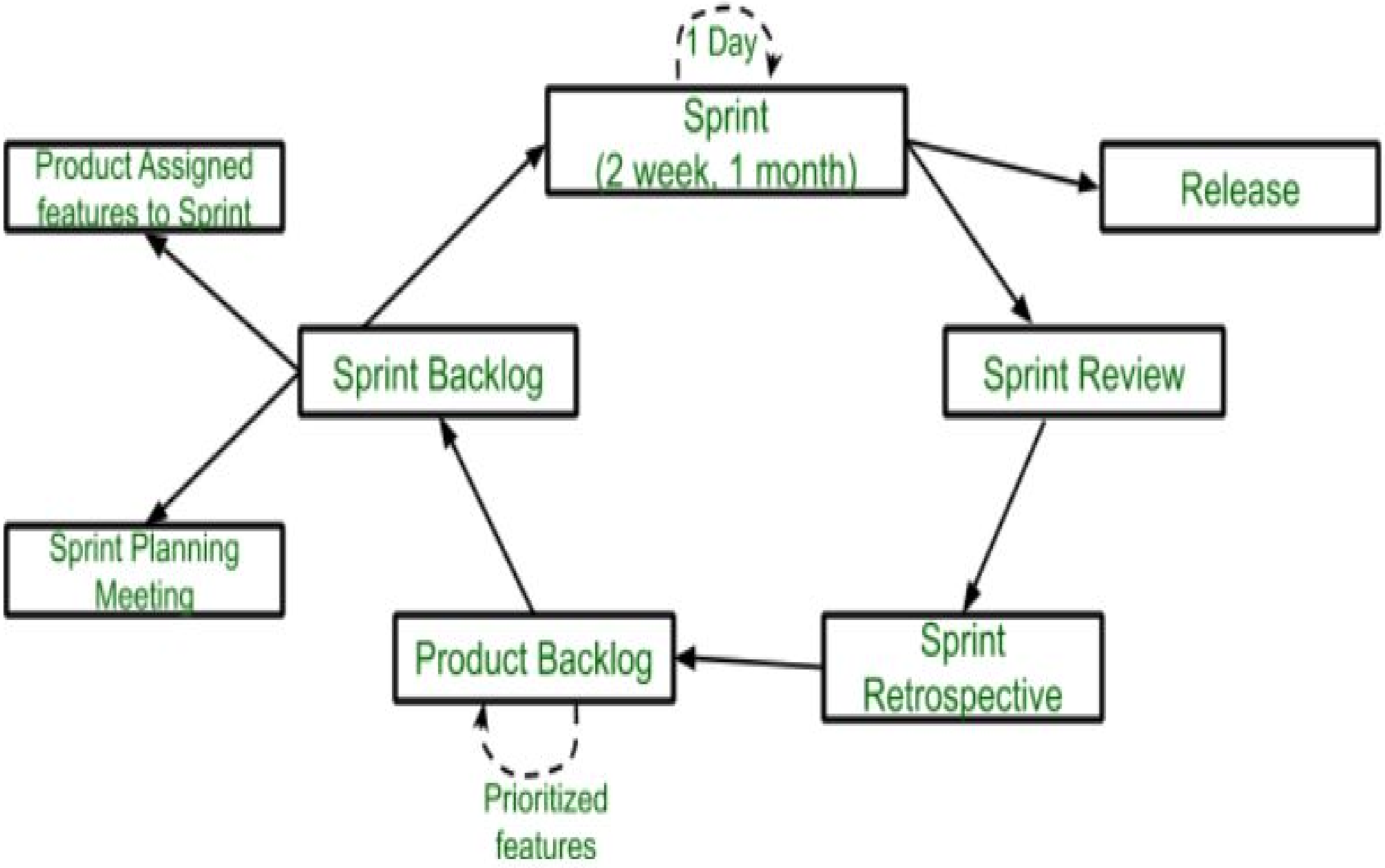
Disadvantages

- The lack of formal documents creates confusion and important decisions taken during different phases can be misinterpreted at any time by different team members.
- It is not suitable for handling complex dependencies.
- 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.

Scrum and XP

- Scrum and Extreme Programming (XP) are both prominent methodologies under the Agile framework, designed to enhance software development processes by promoting flexibility, iterative improvement, and customer satisfaction.

- Scrum is a type of Agile framework. It is a framework within which people can address complex adaptive problems while the productivity and creativity of delivering products are at the highest possible value. Scrum uses an Iterative process.


1 Day
Sprint
(2 week, 1 month)
Product Assigned features to Sprint
Release
Sprint Backlog
Sprint Review
Sprint Planning Meeting
Product Backlog
Sprint Retrospective
Prioritized features

SCRUM PHASES

- *Pre-Game Phase*
- Game Phase (Sprint Execution)
- *Post-Game Phase*

Pre-Game Phase

- **Product Backlog Creation**: List of features, enhancements, bug fixes, etc.
- **Effort Estimation**: Team estimates user stories (e.g., story points).
- **Sprint Planning**:
  - – Selects items from Product Backlog for the Sprint.
  - – Defines a **Sprint Goal**.
  - – Breaks items into tasks.

Game Phase (Sprint Execution)

- **Daily Scrum** (Daily Stand-up):
  - Each team member answers: What did I do? What will I do? Any blockers?
- **Development Work**: Team builds potentially shippable product increments.
- **Sprint Review**: Demonstrate what was built to stakeholders.

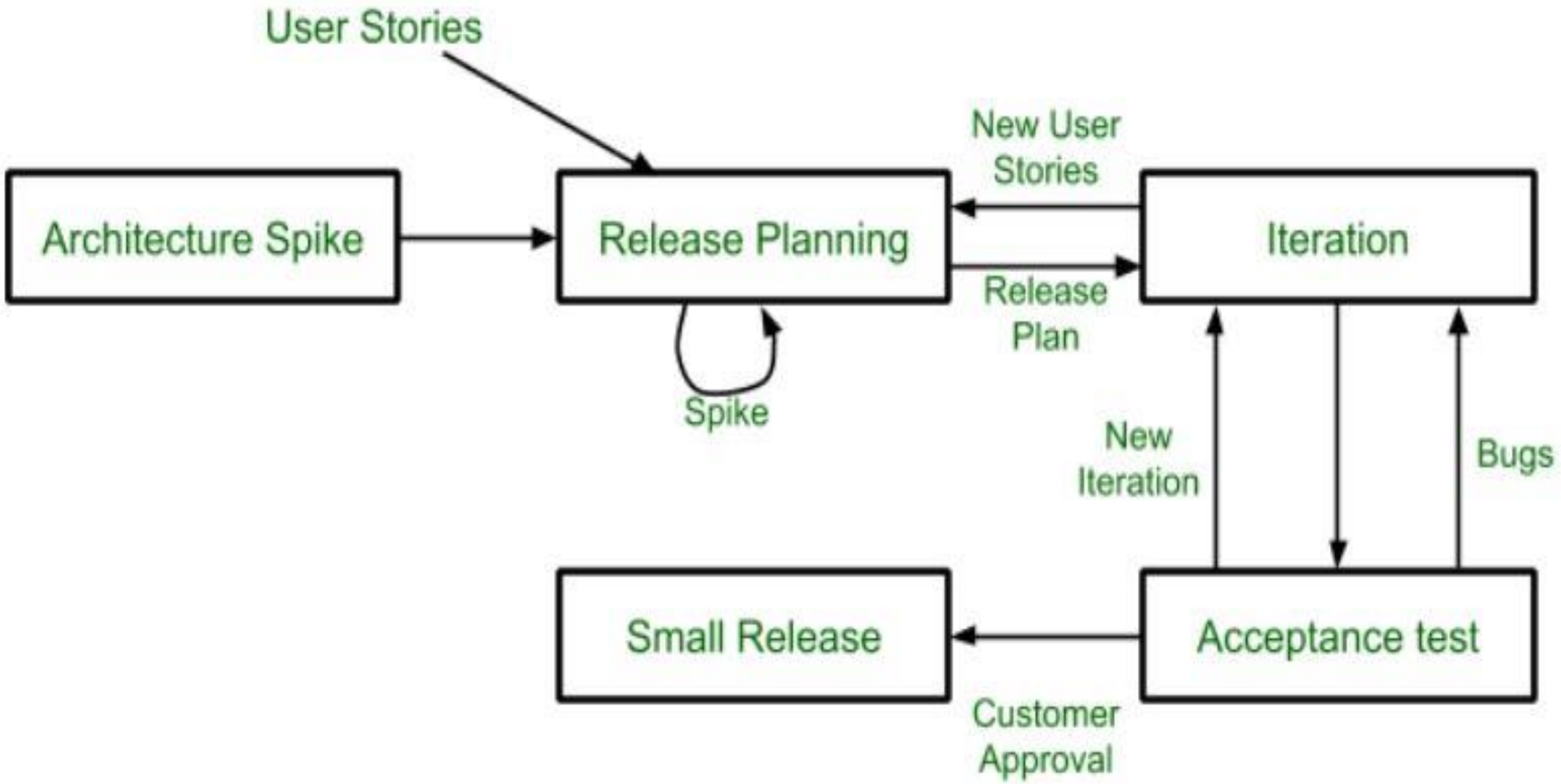
Post-Game Phase

- **Sprint Retrospective**:
  - Inspect the process.
  - Discuss what went well and what can improve.
- **Product Increment Delivery**:
  - Release product or keep it for later integration.

Extreme Programming (XP)

- Extreme programming is one of the most important models of Agile framework. This model emphasizes team-work and customer satisfaction as well. The five basic component of Extreme Programming are:

- Communication
- Simplicity
- Feedback
- Respect
- Courage


User Stories
Architecture Spike
Release Planning
New User Stories
Iteration
Release Plan
Spike
New Iteration
Bugs
Small Release
Acceptance test
Customer Approval

1. *Exploration Phase*

- **Customer defines requirements** via user stories.
- **Team estimates** development effort.
- Spike solutions (small experiments) may be done for unknowns.

2.Planning Phase

- **Release Planning**: Determines what will be delivered and when.
- Prioritizes user stories for the next iteration.
- Creates **Iteration Plan** (usually 1–2 weeks).

3. *Iterations to Release Phase*

- **Short Iterations** (e.g., 1–2 weeks).
- Engineering practices applied:
  - **Pair Programming**
  - **TDD (Test-Driven Development)**
  - **Continuous Integration**
  - **Refactoring**
  - **Collective Code Ownership**
- **Customer Feedback** integrated in each iteration.

- ***4.*** *Productionizing Phase*
- Final polishing, performance testing.
- Hardening the system for release.
- ***5.*** *Maintenance Phase*
- Ongoing support and minor updates after release.
- ***6.*** *Death Phase*
- Project ends, either due to goal completion or cancellation.

Project management

- Project management is the discipline of planning, executing, and overseeing a project to achieve specific goals within defined constraints (such as time, cost, and scope). It typically follows a structured life cycle that helps ensure successful completion.

Stages of Project management

- 5 Stages of Project Management (Project Life Cycle)
- Initiation
- Planning
- Execution
- Monitoring and Controlling
- Closure

1.Initiation

- Goal: Define the project at a broad level and determine its feasibility.
- Key Activities:Develop a business caseConduct a feasibility study
- Identify stakeholders
- Create a project charter
- Deliverable: Approved project charter

Planning

- Goal: Establish the total scope, define objectives, and plan the course of action.
- Key Activities:Define scope, schedule, and budgetCreate a Work Breakdown Structure (WBS)Identify resources, risk management plan, and communication planSet quality and procurement plans
- Deliverable: Project management plan

Execution

- Goal: Deliver the project according to the plan.
- Key Activities:Coordinate people and resourcesManage teams and communicationEnsure quality assuranceConduct procurement and manage stakeholder engagement
- Deliverable: Work performance data, project deliverables

Monitoring and Controlling

- Goal: Track, review, and regulate progress and performance.
- Key Activities:Measure performance (KPIs)Manage scope, schedule, cost, and qualityConduct risk monitoringImplement corrective actions if needed
- Deliverable: Performance reports, change requests, updated project plan

Closure

- Goal: Finalize all activities and formally close the project.
- Key Activities:Obtain client/customer acceptanceRelease resourcesConduct post-project evaluationDocument lessons learnedArchive project documents
- Deliverable: Final project report, archived records, project closure document

4 P's of Project management

- The **4 P's of Project Management** refer to four key components that are essential for successful project delivery. They help guide the structure, execution, and governance of any project.

1. **Project**

- Definition: The specific, temporary endeavor undertaken to create a unique product, service, or result.
- Focus: Scope, objectives, deliverables, constraints.
- Key Questions:What is the goal of the project?What are the expected outcomes?What is the timeline and budget?

2. **Process**

- Definition: The structured approach or methodology used to manage the project from start to finish.Focus: Planning, execution, monitoring, and closure.Examples:WaterfallAgile/ScrumPRINCE2Key Questions:What methodology will be followed?How will tasks be tracked and reported?

3. **People**

- Definition: The individuals or groups involved in the project, including the project team, stakeholders, and clients.Focus: Communication, leadership, roles, responsibilities, collaboration.Key Questions:Who is involved and what are their roles?How will communication be managed?

4. **Performance**

- Definition: The measurement of how well the project meets its objectives in terms of time, cost, quality, and stakeholder satisfaction.Focus: Metrics, KPIs, efficiency, effectiveness.Key Questions:Are we on track with schedule and budget?Are quality standards being met?

Common Project Metrics:

Process and Project Metrics

Metric	Description	Why It Matters
Schedule Variance (SV)	Difference between planned and actual progress	Tracks if you're ahead or behind schedule
Cost Variance (CV)	Difference between budgeted and actual cost	Helps control project spending
Planned Value (PV)	Estimated value of work planned	Shows expected progress
Earned Value (EV)	Estimated value of work actually done	Measures real progress
Actual Cost (AC)	Actual money spent	Used with EV for performance analysis

2. Process Metrics

- Metric	Description	Why It MattersCycle Time	Time to complete one cycle of a process/task	Identifies process delaysLead Time	Time from request initiation to completion	Measures responsivenessThroughput	Number of tasks completed in a time period	Indicates team productivity

railways tracking, arrival time prediction system

- A **Railways Tracking and Arrival Time Prediction System** is typically an **IoT-based or Cloud-integrated Smart Transportation** solution aimed at real-time tracking and predicting train arrivals using a combination of **GPS**, **data analytics**, and **machine learning**.

- **1. Objective**
- To provide real-time tracking of trains and accurate arrival time predictions at stations to improve:
- Passenger convenience
- Operational efficiency
- Safety and coordination

Key Features

Feature	Description
Real-Time Tracking	GPS modules on trains send location data to a central server/cloud.
Arrival Time Prediction	ML algorithms or statistical models predict ETA using current speed, delay, and historical data.
Live Map Display	Users can track trains on a map via mobile/web apps.
Alert System	Sends notifications for arrival, delays, or emergencies.
Data Logging	Maintains travel history for analytics and performance improvement.

3. System Architecture

- **Modules:**
- **Train Unit (IoT device):** GPS + Microcontroller + GSM/LoRa
- **Cloud/Server:** Data collection and prediction algorithms
- **Mobile/Web App:** User interface for tracking and alerts
- **Data Flow:**
  Train → Sends GPS data → Cloud Server → Applies ETA model → Updates User Interface

6. Possible Use Cases

- Indian Railways Train Tracking
- Metro Train ETA App
- School/Corporate Shuttle Tracking
- Freight/Logistics Train Monitoring