Classical Waterfall Model

- The classical waterfall model is the basic software development life cycle model.
- It is very simple but idealistic.
- Earlier this model was very popular but nowadays it is not used.
- But it is very important because all the other software development life cycle models are based on the classical waterfall model.

Why Do We Use the Waterfall Model?

- 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.
- 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 and the project stakeholders need to have a high level of confidence in the outcome.

Features of the Waterfall Model

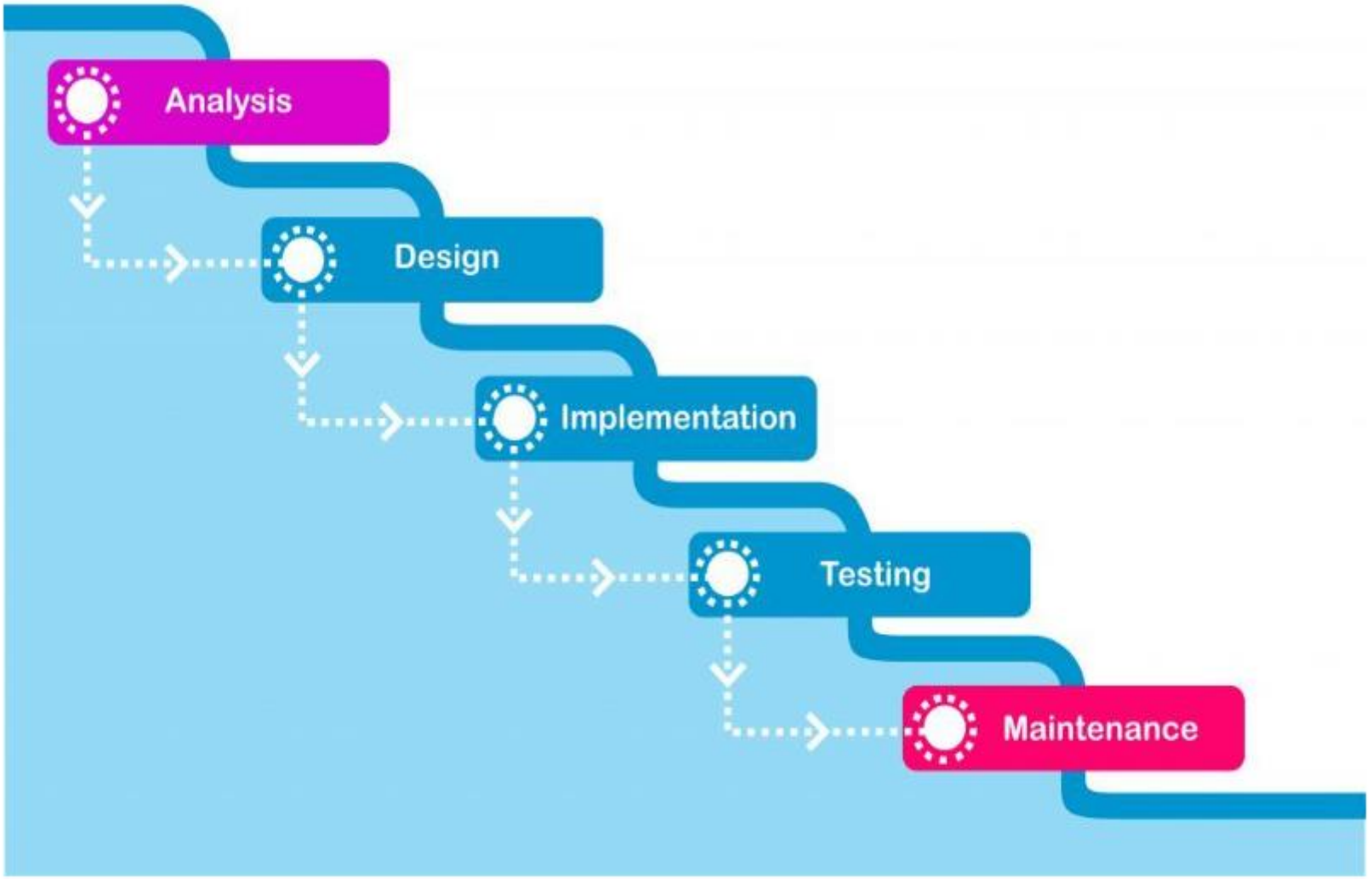
- **Sequential Approach**: The waterfall model involves a sequential approach to software development, where each phase of the project is completed before moving on to the next one.
- **Document-Driven:** The waterfall model relies heavily on documentation to ensure that the project is well-defined and 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.

Features of the Waterfall Model

- **Rigorous Planning**: The waterfall model involves a rigorous planning process, where the project scope, timelines, and deliverables are carefully defined and monitored throughout the project lifecycle.

Overall, the waterfall model is used in situations where there is a need for a highly structured and systematic approach to software development. It can be effective in ensuring that large, complex projects are completed on time and within budget, with a high level of quality and customer satisfaction

WATERFALL



Phases of waterfall Model

Requirement gathering and analysis:

- The first phase involves gathering requirements from stakeholders and analyzing them to understand the scope and objectives of the project.
- **Design:** Once the requirements are understood, the design phase begins. This involves creating a detailed design document that outlines the software architecture, user interface, and system components.

Phases of waterfall Model

- **Implementation:** The implementation phase involves coding the software based on the design specifications. This phase also includes unit testing to ensure that each component of the software is working as expected.
- **Testing:** In the testing phase, the software is tested as a whole to ensure that it meets the requirements and is free from defects

Phases of waterfall Model

- **Implementation:** The implementation phase involves coding the software based on the design specifications. This phase also includes unit testing to ensure that each component of the software is working as expected.
- **Testing:** In the testing phase, the software is tested as a whole to ensure that it meets the requirements and is free from defects

Phases of waterfall Model

- **Deployment:** Once the software has been tested and approved, it is deployed to the production environment.
- **Maintenance:** The final phase of the Waterfall Model is maintenance, which involves fixing any issues that arise after the software has been deployed and ensuring that it continues to meet the requirements over time

Advantages of the 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 the Classical Waterfall Model

- No Feedback Path
- Difficult to accommodate Change Requests.
- No Overlapping of Phases:
- Limited Flexibility:
- Limited Stakeholder Involvement:
- Late Defect Detection:
- Lengthy Development Cycle:
- Not Suitable for Complex Projects:

Applications of Classical Waterfall Model

- Large-scale Software Development Projects: .
- Safety-Critical Systems:
- Government and Defense Projects:
- Projects with well-defined Requirements:
- Projects with Stable Requirements:

The Spiral Model

- **The 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.**

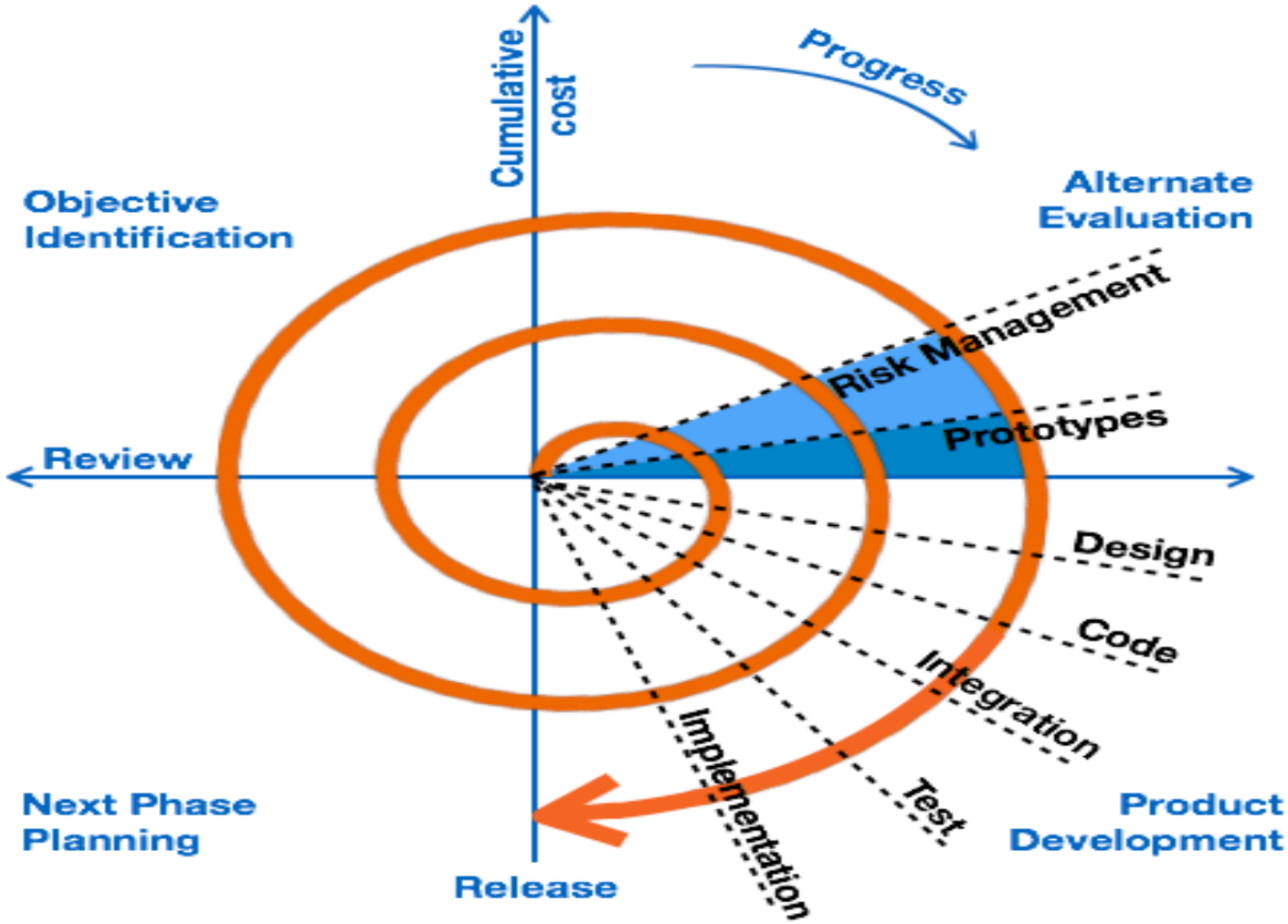
The Spiral Model

- 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 to develop a product using the spiral model.
- The Spiral Model is a **Software Development Life Cycle (SDLC)** model that provides a systematic and iterative approach to software development. 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 of Spiral Model?

- The Spiral Model is a risk-driven model, meaning that the focus is on managing risk through multiple iterations of the software development process.
- It consists of the following phases:

Spiral Model



Spiral Model Application

- The Spiral Model is widely used in the software industry as it is in sync with the natural development process of any product, i.e. learning with maturity which involves minimum risk for the customer as well as the development firms.
- The following pointers explain the typical uses of a Spiral Model −
- When there is a budget constraint and risk evaluation is important.
- For medium to high-risk projects.
- Long-term project commitment because of potential changes to economic priorities as the requirements change with time.
- Customer is not sure of their requirements which is usually the case.
- Requirements are complex and need evaluation to get clarity.
- New product line which should be released in phases to get enough customer feedback.
- Significant changes are expected in the product during the development cycle.

advantages of Spiral Model

- Changing requirements can be accommodated.
- Allows extensive use of prototypes.
- Requirements can be captured more accurately.
- Users see the system early.
- Development can be divided into smaller parts and the risky parts can be developed earlier which helps in better risk management.

The disadvantages Spiral Model

- Management is more complex.
- End of the project may not be known early.
- Not suitable for small or low risk projects and could be expensive for small projects.
- Process is complex
- Spiral may go on indefinitely.
- Large number of intermediate stages requires excessive documentation.

Agile process Model

- The meaning of Agile is swift or versatile. "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.
- Plans regarding the number of iterations, the duration and the scope of each iteration are **clearly defined in advance**.
- Each iteration is considered as a **short time "frame"** in the Agile process model, which typically lasts from one to four weeks.
- The division of the **entire project into smaller parts** helps to minimize the project risk and to reduce the overall project delivery time requirements.
- Each iteration involves a team working through a full software development life cycle including planning, requirements analysis, design, coding, and testing before a working product is demonstrated to the client.

Agile Model

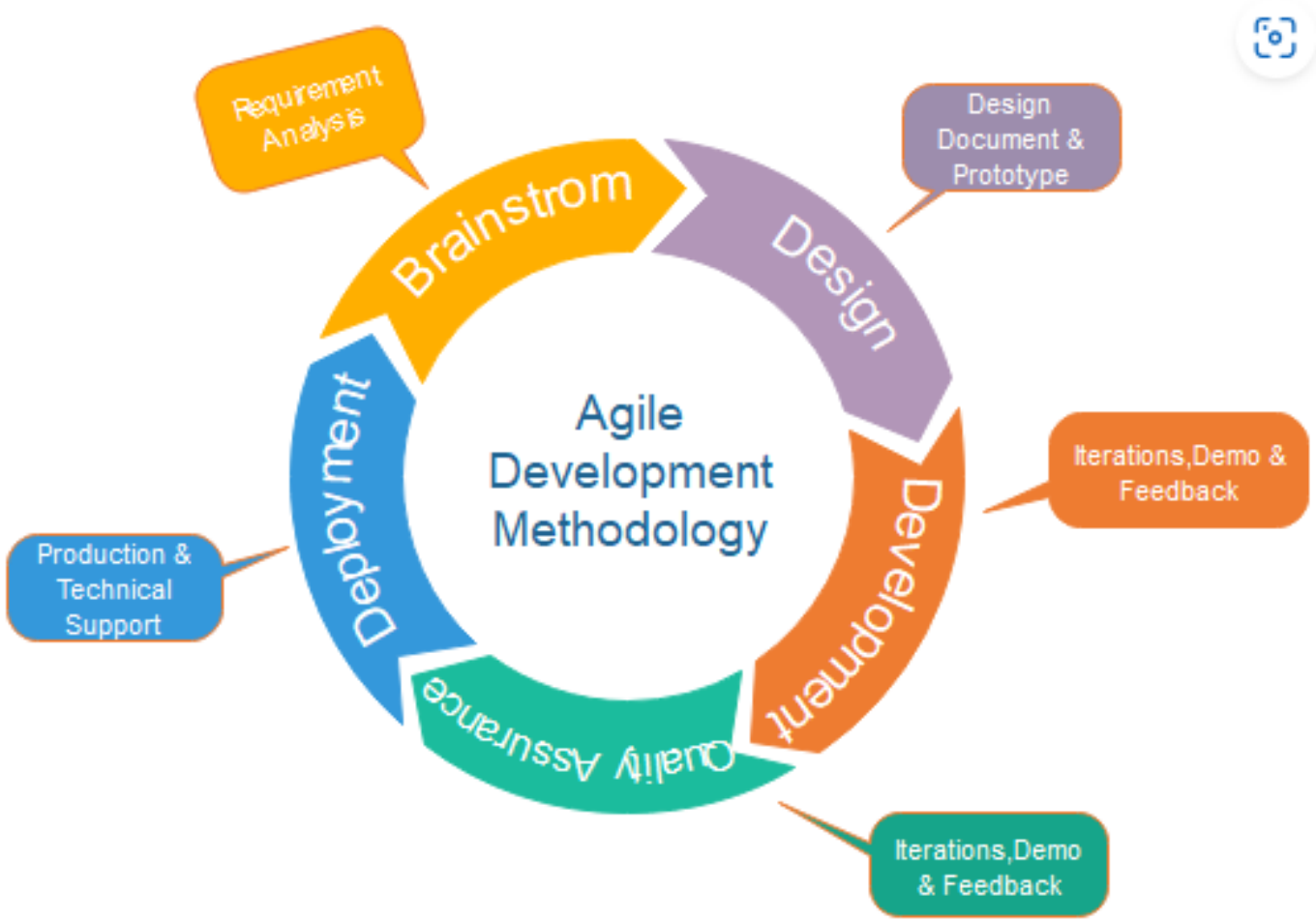


Fig. Agile Model

Phases of Agile Model:

Following are the phases in the Agile model are as follows:

1. Requirements gathering
2. Design the requirements
3. Construction/ iteration
4. Testing/ Quality assurance
5. Deployment
6. Feedback

Phases of Agile Model:

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. Based on this information, you can evaluate technical and economic feasibility.

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.

Construction/ iteration:

When the team defines the requirements, the work begins. Designers and developers start working on their project, which aims to deploy a working product. The product will undergo various stages of improvement, so it includes simple, minimal functionality.

Phases of Agile Model:

Testing:

In this phase, the Quality Assurance team examines the product's performance and looks for the bug.

Deployment:

In this phase, the team issues a product for the user's work environment.

Feedback:

After releasing the product, the last step is feedback. In this, the team receives feedback about the product and works through the feedback.

Agile Testing Methods:

- Scrum
- Crystal
- Dynamic Software Development Method(DSDM)
- Feature Driven Development(FDD)
- Lean Software Development
- eXtreme Programming(XP)

Scrum

SCRUM is an agile development process focused primarily on ways to **manage tasks in team-based development conditions**.

There are three roles in it, and their responsibilities are:

- **Scrum Master:**

  The scrum can **set up the master team**, arrange the meeting and remove obstacles for the process

- **Product owner:**

  The product owner makes the product backlog, prioritizes the delay and is responsible for the distribution of functionality on each repetition.

- **Scrum Team:**

  The team manages its work and organizes the work to **complete the sprint** or **cycle**.

eXtreme Programming(XP)

This type of methodology is used when customers are constantly changing demands or requirements, or when they are not sure about the system's performance.

Crystal:

There are three concepts of this method-

1.Chartering: Multi activities are involved in this phase such as making a development team, performing feasibility analysis, developing plans, etc.

2.Cyclic delivery: under this, two more cycles consist, these are:

- Team updates the release plan.
- Integrated product delivers to the users.

3.Wrap up: According to the user environment, this phase performs deployment, post-deployment.

Dynamic Software Development Method(DSDM):

DSDM is a rapid application development strategy for software development and gives an agile project distribution structure.

The essential features of DSDM are that users must be actively connected, and teams have been given the right to make decisions.

The techniques used in DSDM are:

1. Time Boxing
2. MoSCoW Rules
3. Prototyping

The DSDM project contains seven stages:

1. Pre-project
2. Feasibility Study
3. Business Study
4. Functional Model Iteration
5. Design and build Iteration
6. Implementation
7. Post-project

Feature Driven Development(FDD):

- This method focuses on "**Designing and Building**" **features**. In contrast to other smart methods, FDD describes the small steps of the work that should be obtained separately per function.

Lean Software Development:

- Lean software development methodology follows the principle "**just in time production**." The lean method indicates the **increasing speed of software development and reducing costs**. Lean development can be summarized in **seven phases**

1. Eliminating Waste
2. Amplifying learning
3. Defer commitment (deciding as late as possible)
4. Early delivery
5. Empowering the team
6. Building Integrity
7. Optimize the whole

When to use the Agile Model?

- When frequent changes are required.
- When a highly qualified and experienced team is available.
- When a customer is ready to have a meeting with a software team all the time.
- When project size is small.

Advantage(Pros) of Agile Method:

- 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(Cons) of Agile Model:

- 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.