SOFTWARE DEVELOPMENT LIFE CYCLE (SDLC)

What is Software Development Life Cycle (SDLC)?

- Software development lifecycle (SDLC) is a framework that defines the steps involved in the development of software. It covers the detailed plan for building, deploying and maintaining the software.
- SDLC defines the complete cycle of development i.e. all the tasks involved in gathering a requirement for the maintenance of a Product.

SDLC Process

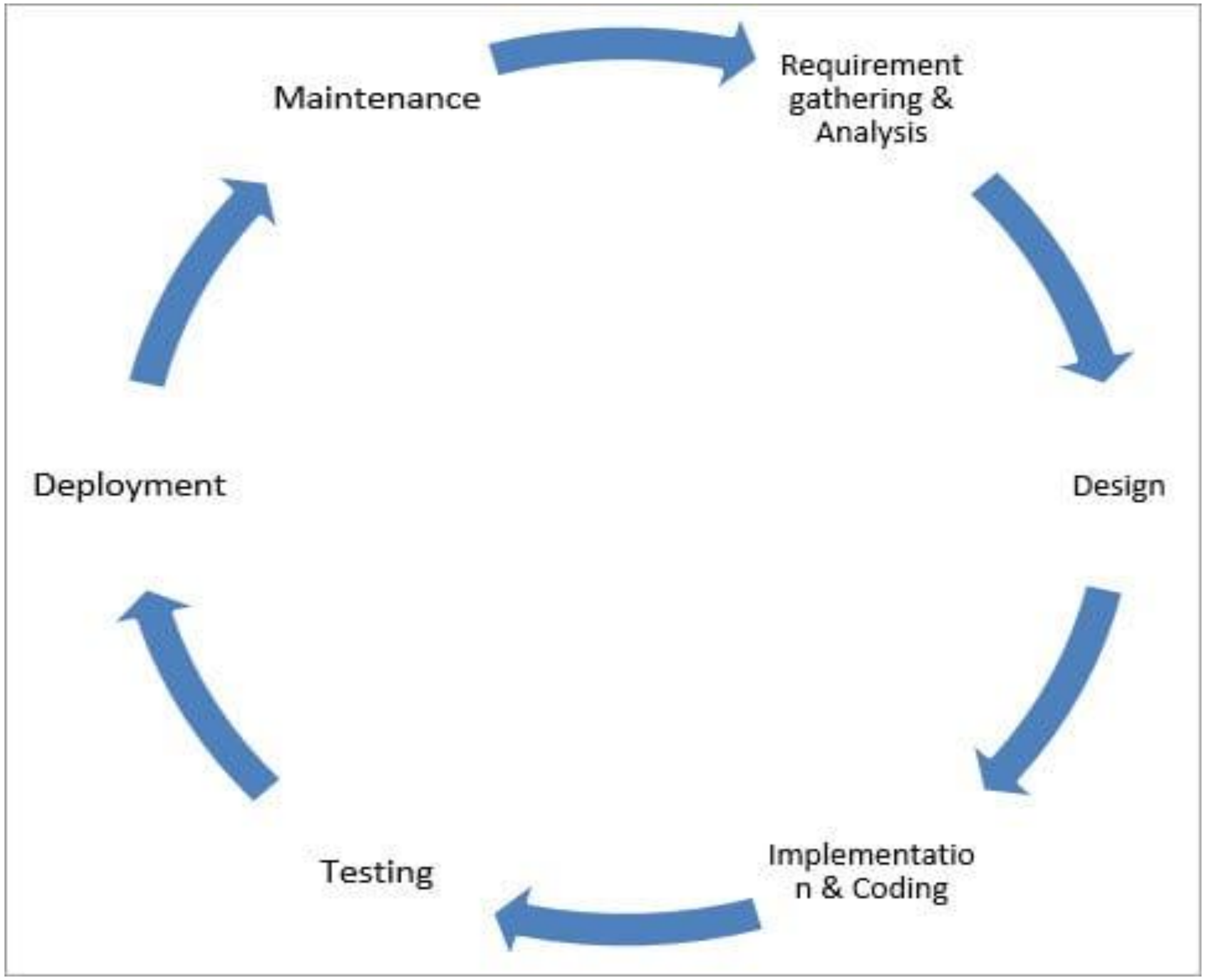
- SDLC is a process which defines the various stages involved in the development of software for delivering a high-quality product. SDLC stages cover the complete life cycle of a software i.e. from inception to retirement of the product.
- Adhering to the SDLC process leads to the development of the software in a systematic and disciplined manner.

Purpose of SDLC

- Purpose of SDLC is to deliver a high-quality product which is as per the customer's requirement.
- SDLC has defined its phases as, Requirement gathering, Designing, Coding, Testing, and Maintenance. It is important to adhere to the phases to provide the Product in a systematic manner.
- For Example, A software has to be developed and a team is divided to work on a feature of the product and is allowed to work as they want. One of the developers decides to design first whereas the other decides to code first and the other on the documentation part. This will lead to project failure because of which it is necessary to have a good knowledge and understanding among the team members to deliver an expected product.

SDLC Cycle

SDLC Cycle represents the process of developing software.



SDLC Phases

Given below are the various phases of SDLC:

- Requirement gathering and analysis
- Design
- Implementation or coding
- Testing
- Deployment
- Maintenance

Requirement Gathering and Analysis

During this phase, all the relevant information is collected from the customer to develop a product as per their expectation. Any ambiguities must be resolved in this phase only.

Business analyst and Project Manager set up a meeting with the customer to gather all the information like what the customer wants to build, who will be the end user, what is the purpose of the product. Before building a product a core understanding or knowledge of the product is very important.

For Example, A customer wants to have an application which involves money transactions. In this case, the requirement has to be clear like what kind of transactions will be done, how it will be done, in which currency it will be done, etc.

Once the requirement gathering is done, an analysis is done to check the feasibility of the development of a product. In case of any ambiguity, a call is set up for further discussion.

Once the requirement is clearly understood, SRS (Software Requirement Specification) document is created. This document should be thoroughly understood by the developers and also should be reviewed by the customer for future reference.

Design

In this phase, the requirement gathered in the SRS document is used as an input and software architecture that is used for implementing system development is derived.

Implementation or Coding

Implementation/Coding starts once the developer gets the Design document. The Software design is translated into source code. All the components of the software are implemented in this phase.

Testing

Testing starts once the coding is complete and the modules are released for testing. In this phase, the developed software is tested thoroughly and any defects found are assigned to developers to get them fixed.

Retesting, regression testing is done till the point at which the software is as per the customer’s expectation. Testers refer SRS document to make sure that the software is as per the customer’s standard.

Deployment

Once the product is tested, it is deployed in the production environment or first UAT (User Acceptance testing) is done depending on the customer expectation.

In the case of UAT, a replica of the production environment is created and the customer along with the developers does the testing. If the customer finds the application as expected, then sign off is provided by the customer to go live.

Maintenance

After the deployment of a product on the production environment, maintenance of the product i.e. if any issue comes up and needs to be fixed or any enhancement is to be done is taken care by the developers.

WHAT IS SDLC PROCESS MODELS?

- A software life cycle model is a descriptive representation of the software development cycle. SDLC models might have a different approach but the basic phases and activity remain the same for all the models.
- SDLC models have been created by software development experts, universities, and standards organizations to solve some repeated issue or to enhance other models.
- Each process model follows series of steps unique to its type, in order to ensure success in process of software development.

WHY USING A PROCESS MODEL

Project Planning

What should be or shouldn't be built

Define the terminologies, activities and deliverables

Proper Documentation

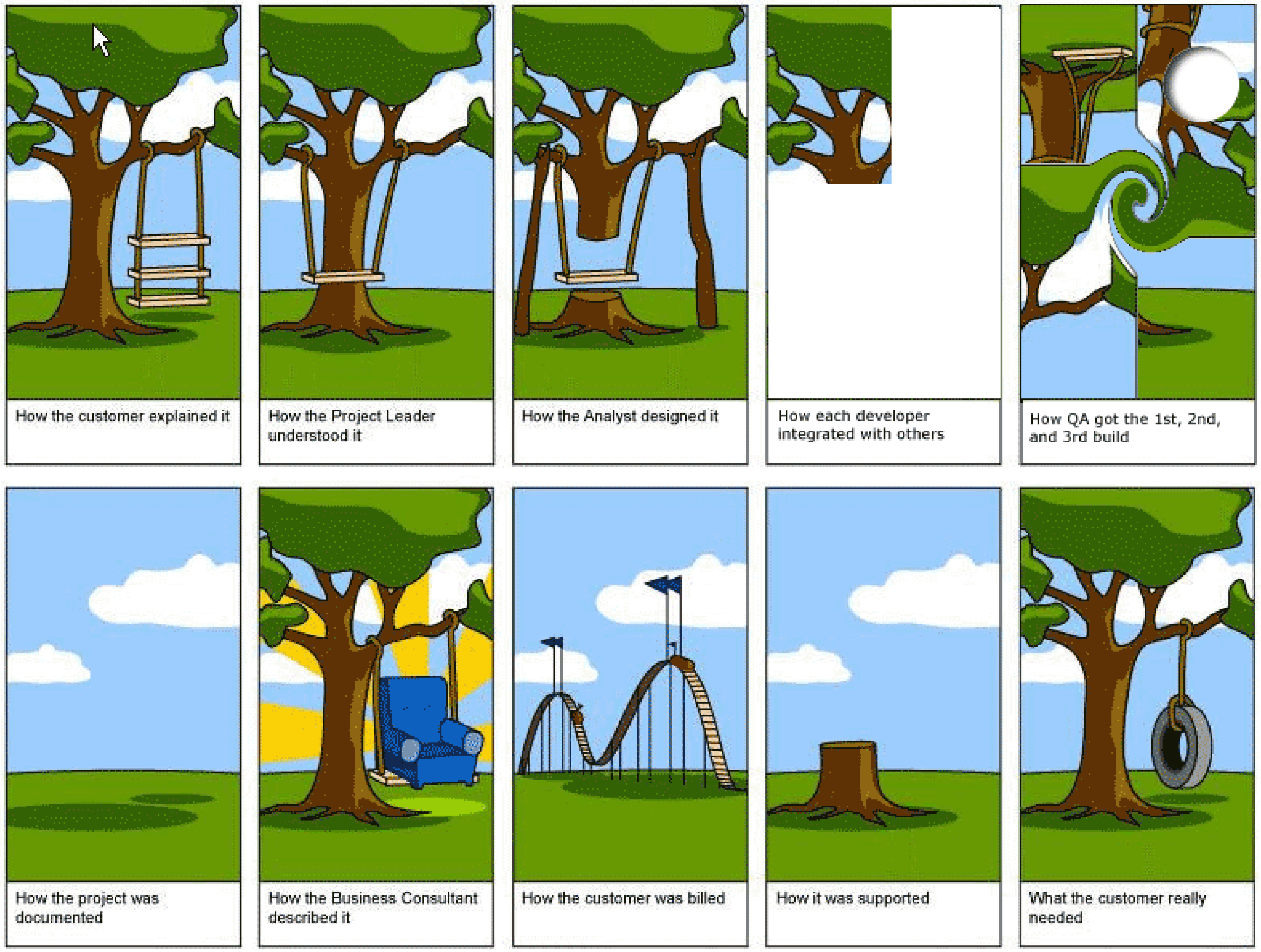
Align project progress visibility with stakeholders

THE RIGHT CHOICE

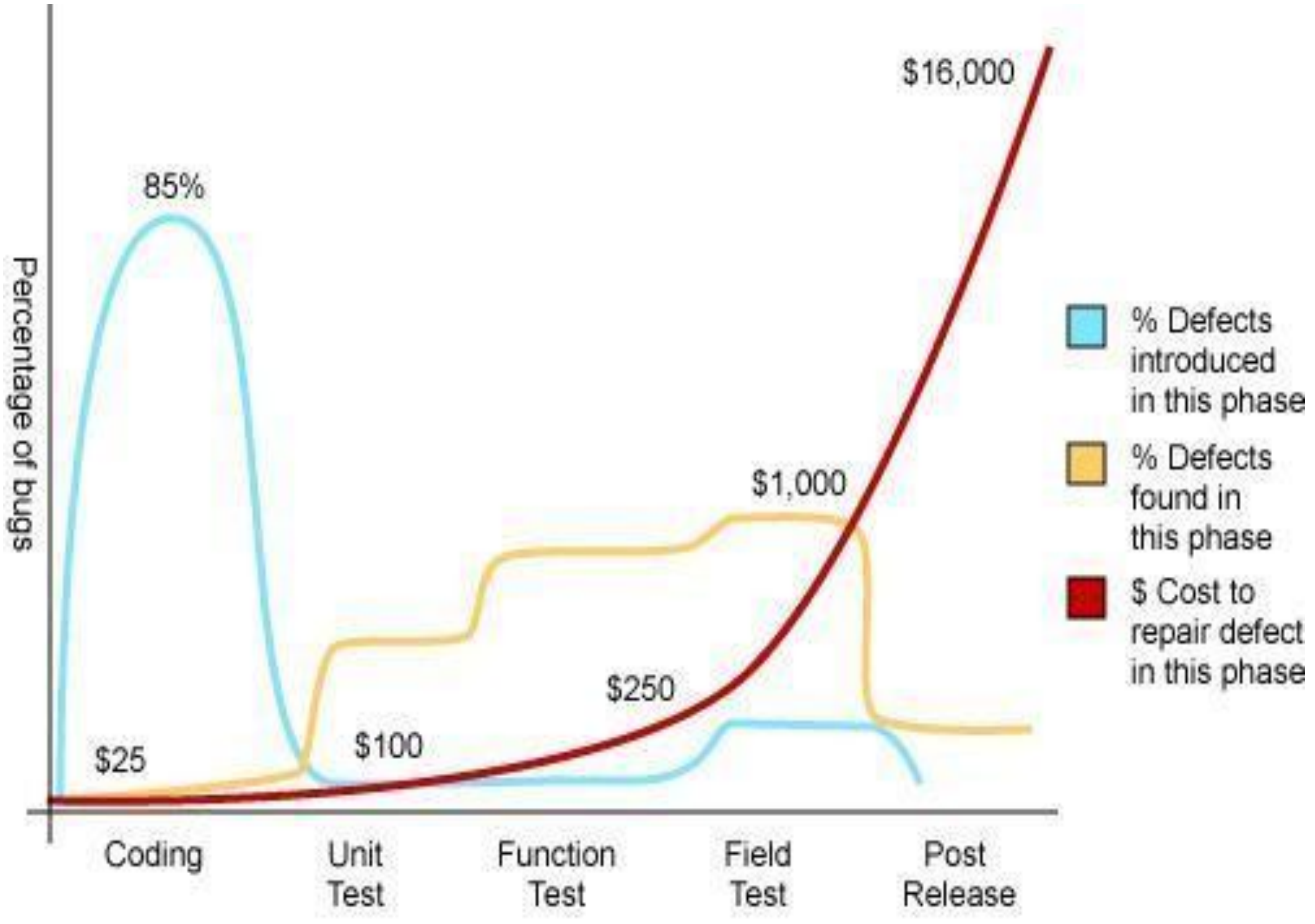
We need to choose the right SDLC based on project context

- Improve customer alignment and relations
- Development speed (time to market)
- Increase projects success rate
- Improve software quality
- Eliminating management overhead
- Decrease implementation risk

PERSPECTIVES


How the customer explained it
How the Project Leader understood it
How the Analyst designed it
How each developer integrated with others
How QA got the 1st, 2nd, and 3rd build
How the project was documented
How the Business Consultant described it
How the customer was billed
How it was supported
What the customer really needed

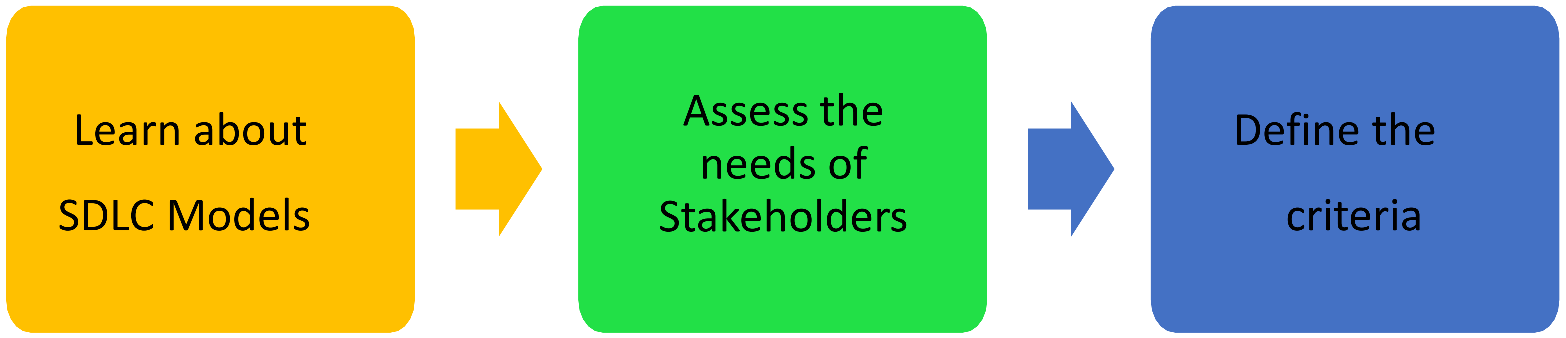
THE COST OF DEFECTS



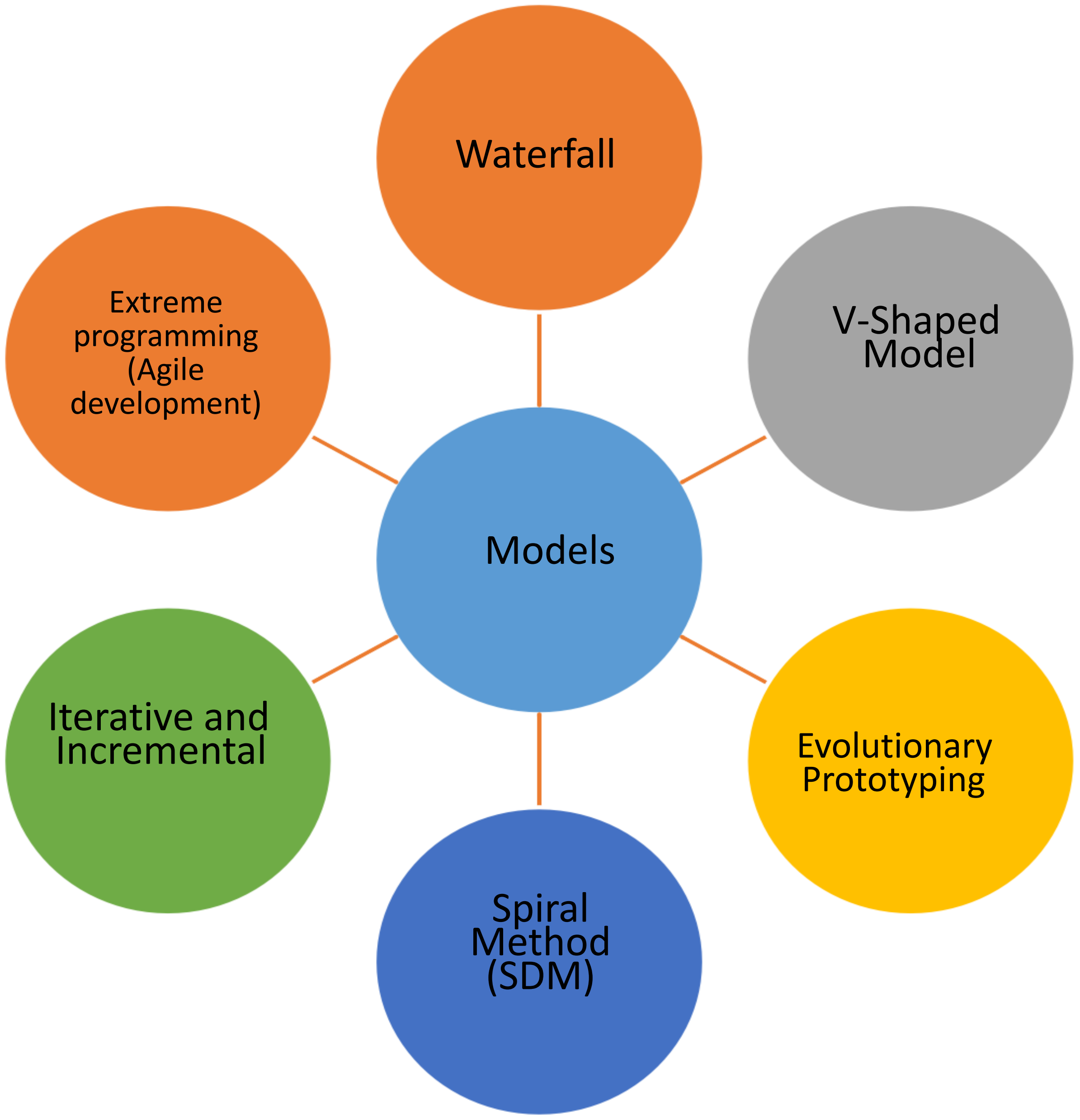
Source: Applied Software Measurement, Capers Jones, 1996

HOW TO SELECT THE RIGHT SDLC

Selecting the right SDLC is a process in itself that organization can implement internally or consult for. There are some steps to get the right selection



MOST COMMON MODELS



Waterfall Model

Waterfall model is the very first model that is used in SDLC. It is also known as the linear sequential model. In this model, the outcome of one phase is the input for the next phase. Development of the next phase starts only when the previous phase is complete.

First, **Requirement gathering and analysis** is done. Once the requirement is freeze then only the System Design can start. Here in, the SRS document created is the output for the Requirement phase and it acts as an input for the System Design.

In **System Design** Software architecture and Design, documents which act as an input for the next phase are created i.e. Implementation and coding.

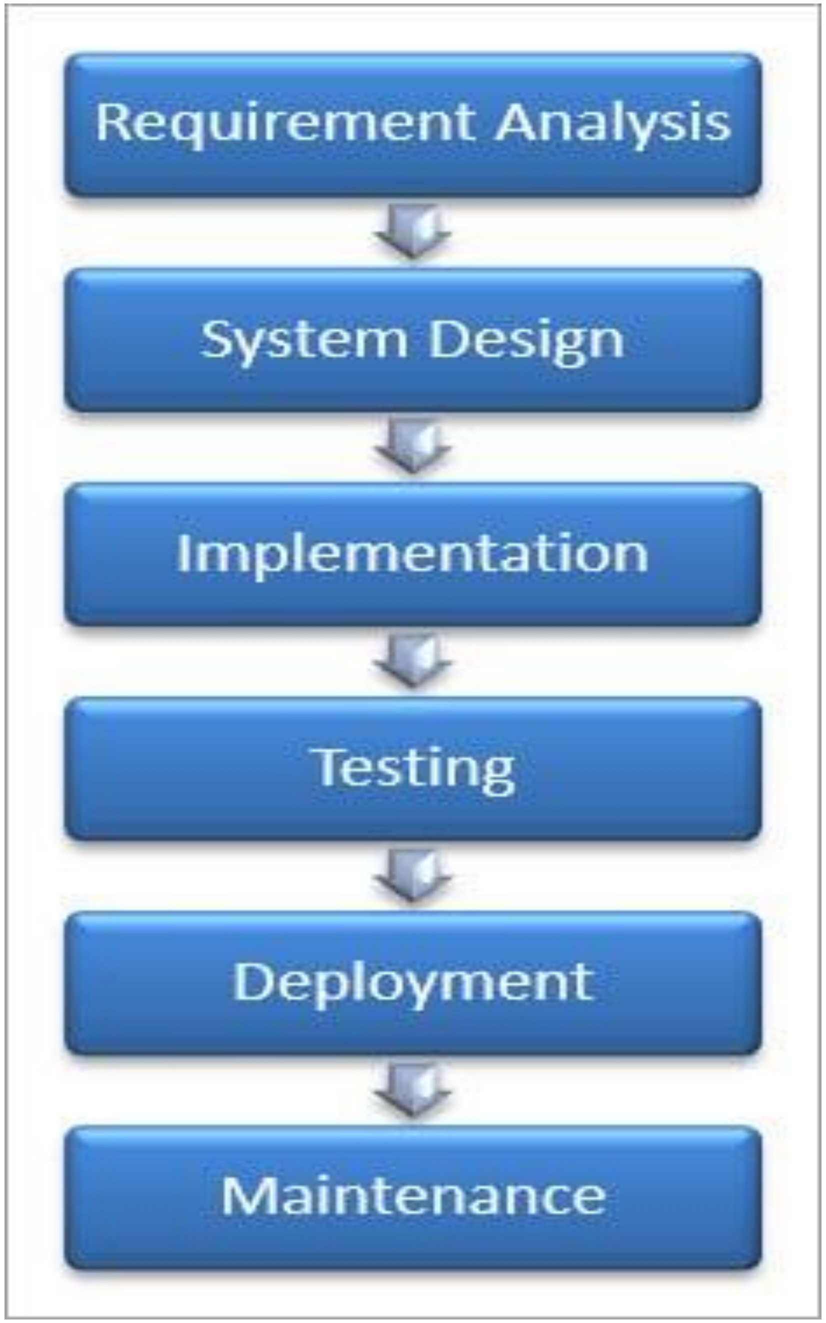
In the **Implementation phase**, coding is done and the software developed is the input for next phase i.e. testing.

In the **testing phase**, the developed code is tested thoroughly to detect the defects in the software. Defects are logged into the defect tracking tool and are retested once fixed. Bug logging, Retest, Regression testing goes on until the time the software is in go-live state.

In the **Deployment phase**, the developed code is moved into production after the sign off is given by the customer.

Any issues in the production environment are resolved by the developers which come under **maintenance**.

Waterfall Model



Advantages of the Waterfall Model:

Waterfall model is the **simple** model which can be easily understood and is the one in which all the phases are done step by step.

Deliverables of each phase are well defined, and this leads to **no complexity** and makes the project **easily manageable**.

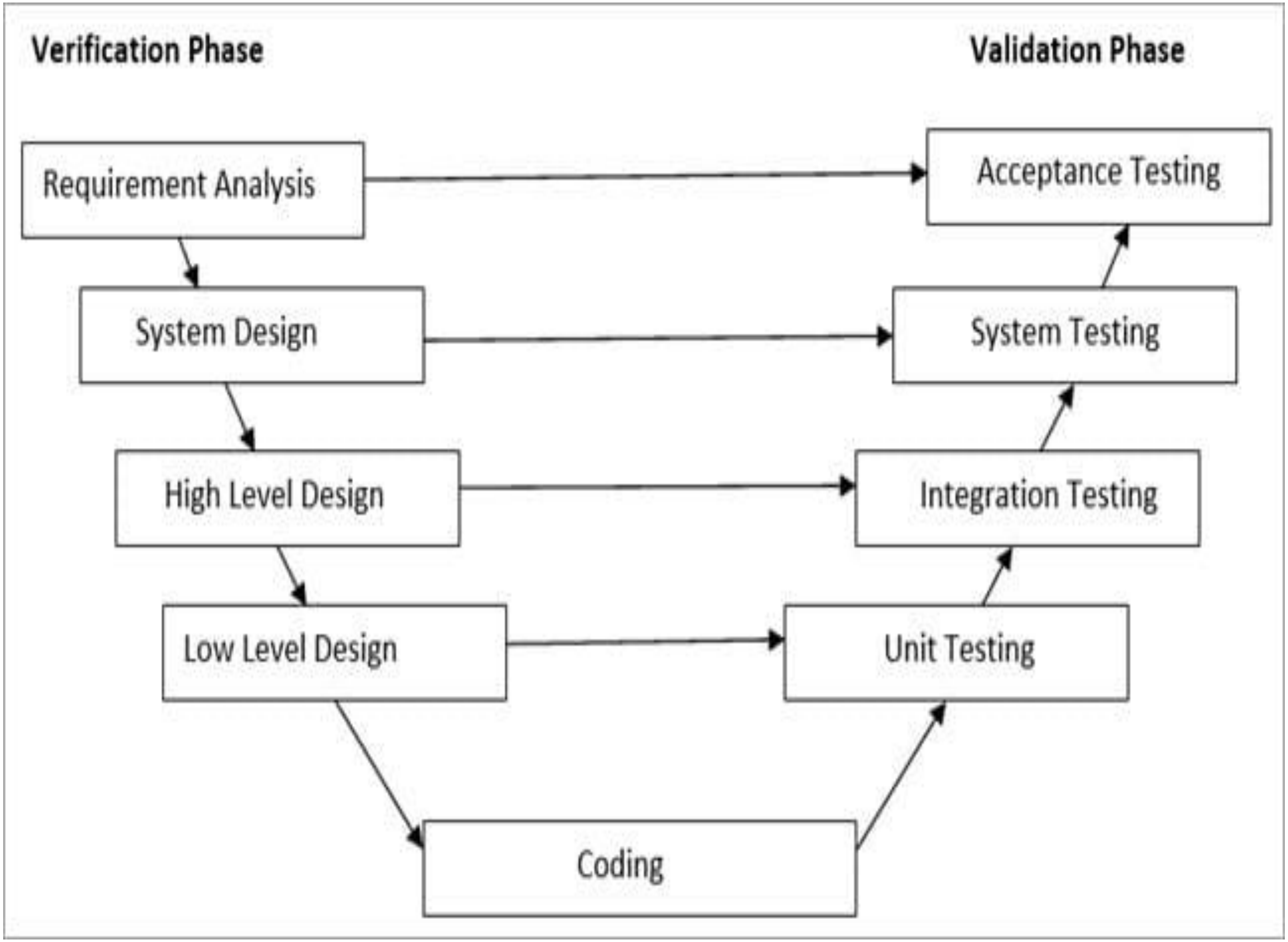
Disadvantages of Waterfall model:

Waterfall model **is time-consuming** & cannot be used in the short duration projects as in this model a new phase cannot be started till the ongoing phase is completed.

Waterfall model **cannot be used for the projects which have uncertain requirement** or wherein the requirement keeps on changing as this model expects the requirement to be clear in the requirement gathering and analysis phase itself and any change in the later stages would lead to cost higher as the changes would be required in all the phases.

V-Shaped Model:

V- Model is also known as **Verification and Validation Model**. In this model Verification & Validation goes hand in hand i.e. development and testing goes parallel. **V model and waterfall model are the same except that the test planning and testing start at an early stage in V-Model.**


Verification Phase
Validation Phase
Requirement Analysis
Acceptance Testing
System Design
System Testing
High Level Design
Integration Testing
Low Level Design
Unit Testing
Coding

Verification Phase:

(i) Requirement Analysis:

In this phase, all the required information is gathered & analyzed. Verification activities include reviewing the requirements.

(ii) System Design:

Once the requirement is clear, a system is designed i.e. architecture, components of the product are created and documented in a design document.

(iii) High-Level Design:

High-level design defines the architecture/design of modules. It defines the functionality between the two modules.

(iv) Low-Level Design:

Low-level Design defines the architecture/design of individual components.

(v) Coding:

Code development is done in this phase.

Validation Phase:

(i) Unit Testing:

Unit testing is performed using the unit test cases that are designed and is done in the Low-level design phase. Unit testing is performed by the developer itself. It is performed on individual components which lead to early defect detection.

(ii) Integration Testing:

Integration testing is performed using integration test cases in High-level Design phase. Integration testing is the testing that is done on integrated modules. It is performed by testers.

(iii) System Testing:

System testing is performed in the System Design phase. In this phase, the complete system is tested i.e. the entire system functionality is tested.

(iv) Acceptance Testing:

Acceptance testing is associated with the Requirement Analysis phase and is done in the customer's environment.

Advantages of V – Model:

- It is the simple and easily understandable model.
- V –model approach is good for smaller projects wherein the requirement is defined and it freezes in the early stage.
- It is a systematic and disciplined model which results in a high-quality product.

Disadvantages of V-Model:

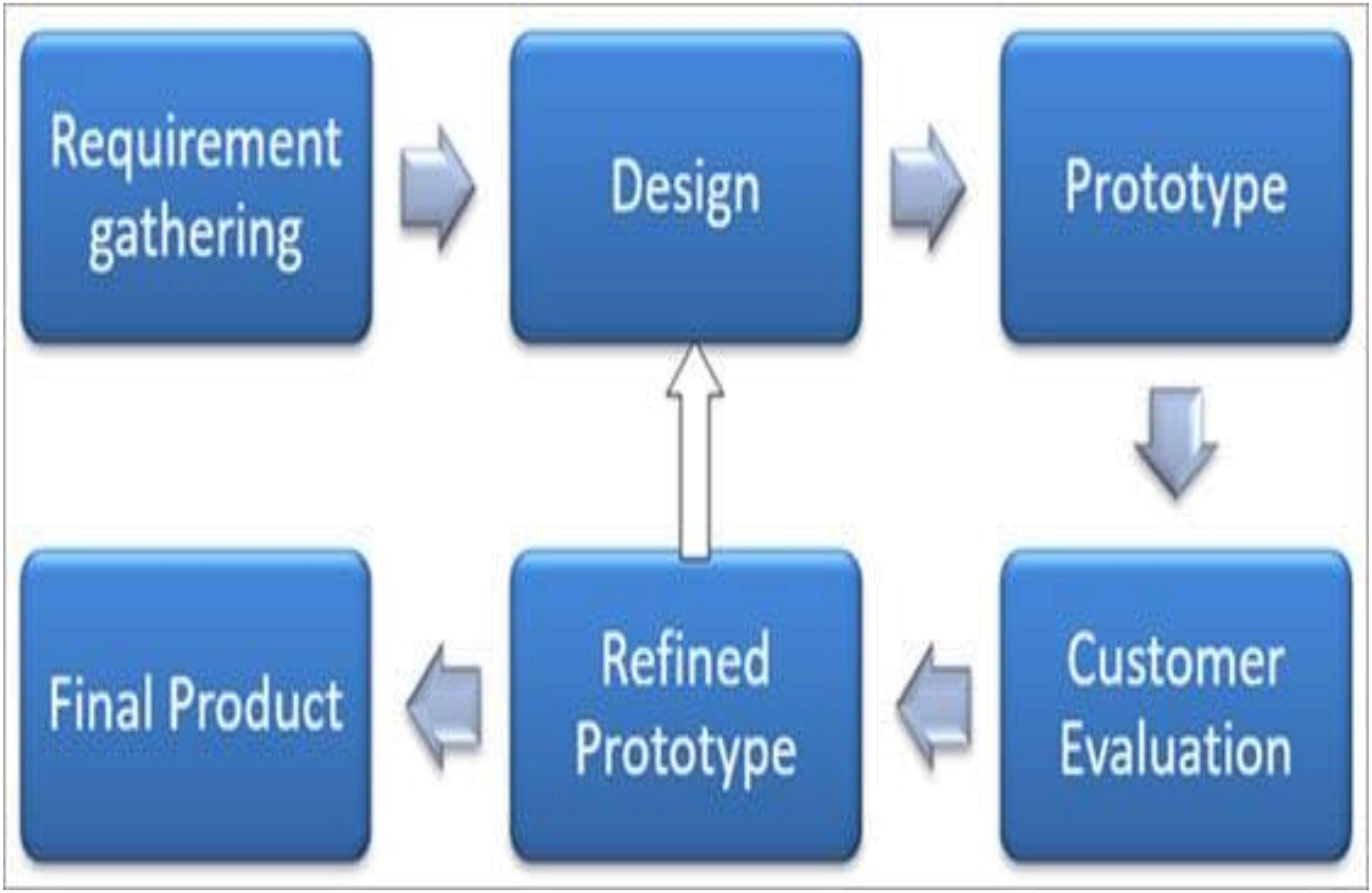
- V-shaped model is not good for ongoing projects.
- Requirement change at the later stage would cost too high.

Prototype Model

The prototype model is a model in which the prototype is developed prior to actual software.

Prototype models have limited functional capabilities and inefficient performance when compared to the actual software. Dummy functions are used to create prototypes. This is a valuable mechanism for understanding the customers' needs.

Software prototypes are built prior to the actual software to get valuable feedback from the customer. Feedbacks are implemented and the prototype is again reviewed by the customer for any change. This process goes on till the model is accepted by the customer.


Requirement gathering
Design
Prototype
Customer Evaluation
Refined Prototype
Final Product

Once the requirement gathering is done, the quick design is created and the prototype which is presented to the customer for evaluation is built.

Customer feedback and the refined requirement is used to modify the prototype and is again presented to the customer for evaluation. Once the customer approves the prototype, it is used as a requirement for building the actual software. The actual software is build using the Waterfall model approach.

Advantages of Prototype Model:

Prototype model reduces the cost and time of development as the defects are found much earlier.

Missing feature or functionality or a change in requirement can be identified in the evaluation phase and can be implemented in the refined prototype.

Involvement of a customer from the initial stage reduces any confusion in the requirement or understanding of any functionality.

Disadvantages of Prototype Model:

Since the customer is involved in every phase, the customer can change the requirement of the end product which increases the complexity of the scope and may increase the delivery time of the product.

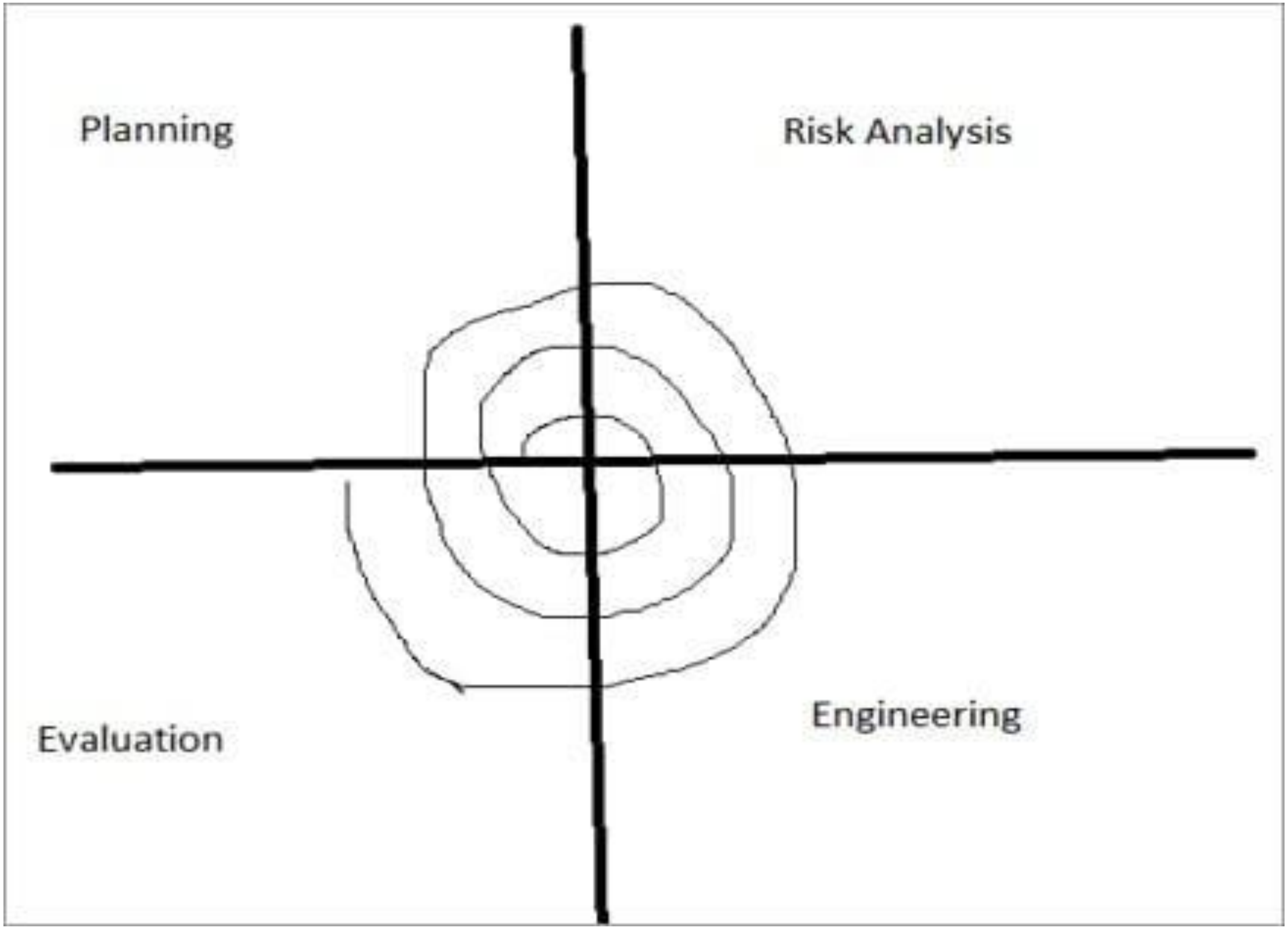
Spiral Model

The Spiral Model includes iterative and prototype approach.

Spiral model phases are followed in the iterations. The loops in the model represent the phase of the SDLC process i.e. the innermost loop is of requirement gathering & analysis which follows the Planning, Risk analysis, development, and evaluation. Next loop is Designing followed by Implementation & then testing.

Spiral Model has four phases:

- Planning
- Risk Analysis
- Engineering
- Evaluation


Planning
Risk Analysis
Evaluation
Engineering

(i) Planning:

Planning phase includes requirement gathering wherein all the required information is gathered from the customer and is documented. Software requirement specification document is created for the next phase.

(ii) Risk Analysis:

In this phase, the best solution is selected for the risks involved and analysis is done by building the prototype.

For Example, the risk involved in accessing the data from a remote database can be that the data access rate might be too slow. The risk can be resolved by building a prototype of the data access subsystem.

(iii) Engineering:

Once the risk analysis is done, coding and testing are done.

(iv) Evaluation:

Customer evaluates the developed system and plans for the next iteration.

Advantages of Spiral Model:

- Risk Analysis is done extensively using the prototype models.
- Any enhancement or change in the functionality can be done in the next iteration.

Disadvantages of Spiral Model:

- The spiral model is best suited for large projects only.
- Cost can be high as it might take large no of iterations which can lead to high time to reach the final product.

Iterative Incremental Model

The iterative incremental **model divides the product into small chunks**.

For Example, Feature to be developed in the iteration is decided and implemented. Each iteration goes through the **phases namely Requirement Analysis, Designing, Coding, and Testing**. Detailed planning is not required in iterations.

Once the iteration is completed, a product is verified and is delivered to the customer for their evaluation and feedback. **Customer's feedback is implemented in the next iteration along with the newly added feature.**

Hence, the product increments in terms of features and once the iterations are completed the final build holds all the features of the product.

Phases of Iterative & Incremental Development Model:

- Inception phase
- Elaboration Phase
- Construction Phase
- Transition Phase

(i) Inception Phase:

Inception phase includes the requirement and scope of the Project.

(ii) Elaboration Phase:

In the elaboration phase, the working architecture of a product is delivered which covers the risk identified in the inception phase and also fulfills the non-functional requirements.

(iii) Construction Phase:

In the Construction phase, the architecture is filled in with the code which is ready to be deployed and is created through analysis, designing, implementation, and testing of the functional requirement.

(iv) Transition Phase:

In the Transition phase, the product is deployed in the Production environment.

Advantages of Iterative & Incremental Model:

- Any change in the requirement can be easily done and would not cost as there is a scope of incorporating the new requirement in the next iteration.
- Risk is analyzed & identified in the iterations.
- Defects are detected at an early stage.
- As the product is divided into smaller chunks it is easy to manage the product.

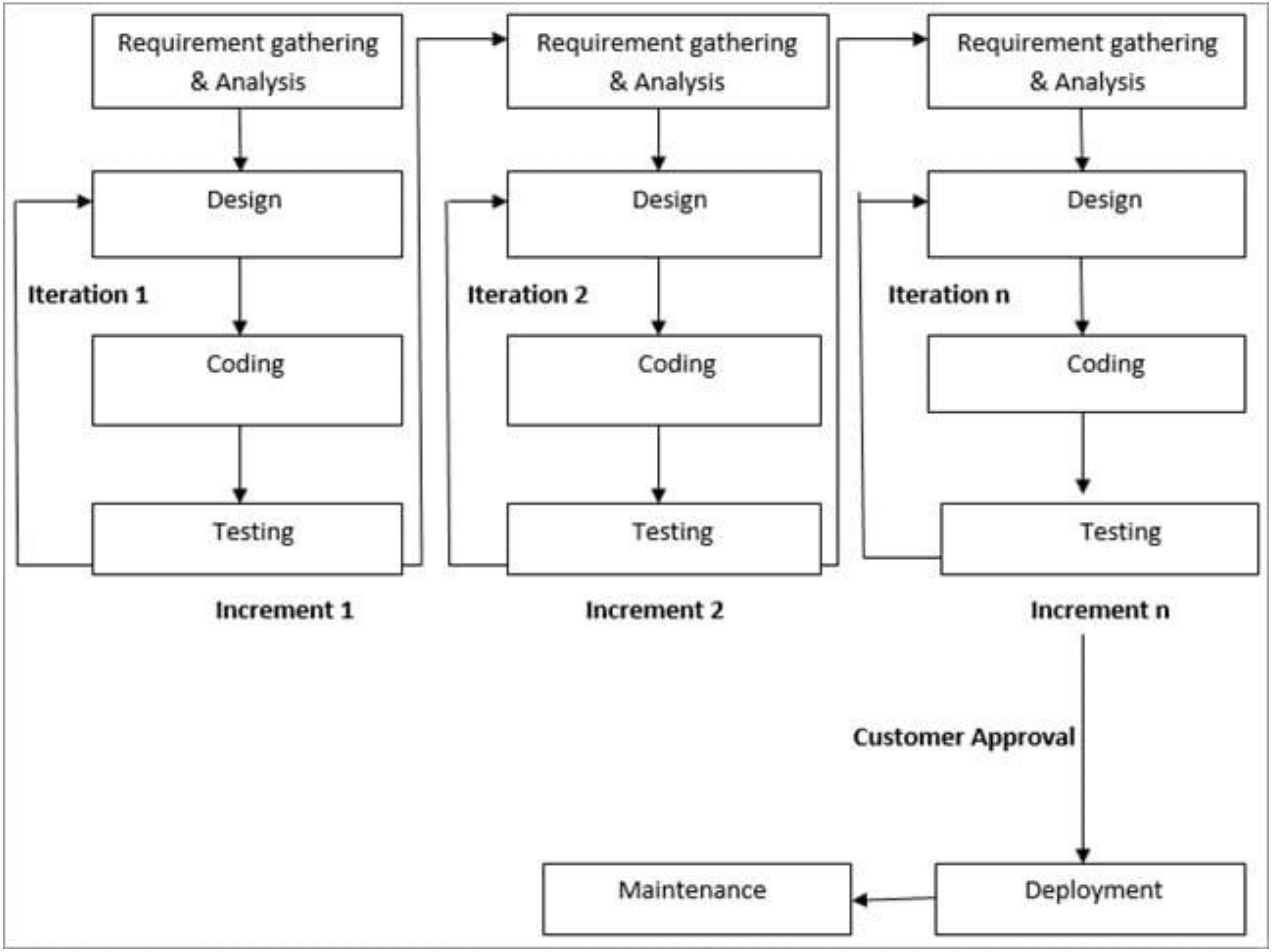
Disadvantages of Iterative & Incremental Model:

- Complete requirement and understanding of a product are required to break down and build incrementally.

Agile Model

- Agile Model is a combination of the Iterative and incremental model. This model focuses more on flexibility while developing a product rather than on the requirement.
- In Agile, a product is broken into small incremental builds. It is not developed as a complete product in one go. Each build increments in terms of features. The next build is built on the previous functionality.

- In agile iterations are termed as sprints. Each sprint lasts for2-4 weeks. At the end of each sprint, the product owner verifies the product and after his approval, it is delivered to the customer.
- Customer feedback is taken for improvement and his suggestions and enhancement are worked on in the next sprint. Testing is done in each sprint to minimize the risk of any failures.


Requirement gathering & Analysis
Design
Coding
Testing
Iteration 1
Increment 1
Requirement gathering & Analysis
Design
Coding
Testing
Iteration 2
Increment 2
Requirement gathering & Analysis
Design
Coding
Testing
Iteration n
Increment n
Customer Approval
Deployment
Maintenance

Advantages of Agile Model:

- It allows more flexibility to adapt to the changes.
- The new feature can be added easily.
- Customer satisfaction as the feedback and suggestions are taken at every stage.

Disadvantages:

- Lack of documentation.
- Agile needs experienced and highly skilled resources.
- If a customer is not clear about how exactly they want the product to be, then the project would fail.