




## Unit-3

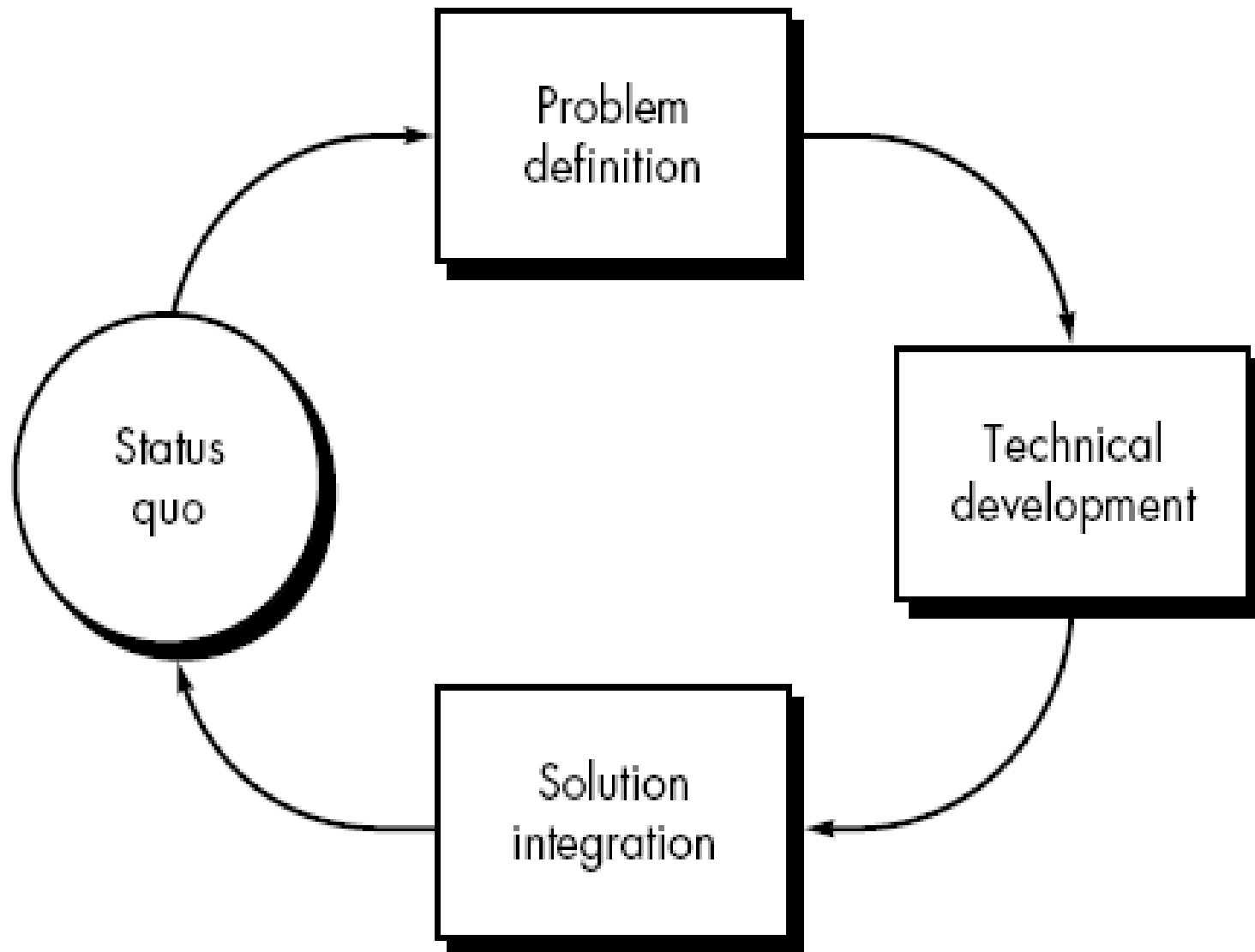
# Software Development Life Cycle Models, Automated Testing



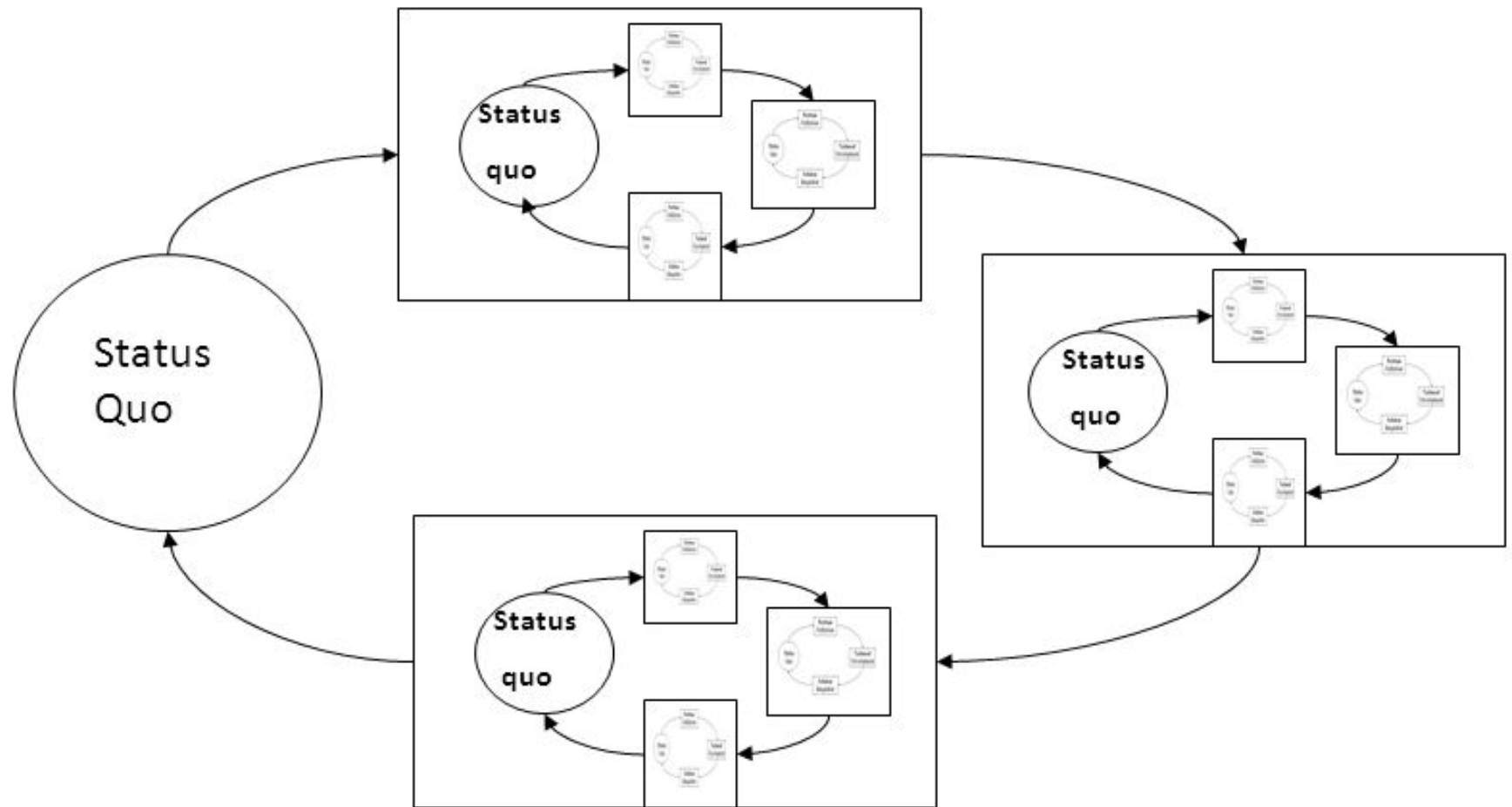
# Software Development Models

- A Software engineer must prepare development plan that covers the process, methods, and tools to solve problems.
- This plan is referred as a process model or a software engineering paradigm.
- A process model is chosen- based on the nature of the project and application, the methods and tools to be used, and the controls and deliverables.

- 
- All software development can be specified as a problem solving loop as shown below. Four distinct stages are :
  - **Status Quo**
  - **Problem Definition**
  - **Technical Development**
  - **Solution Integration**





# Software Process Models




The phases within phases of the problem solving loop



- 
- Status quo is the current state of problem.
  - Problem definition identifies the specific problem to be solved.
  - Technical development solves through some technology and solution integration delivers the result(e.g..document, program, data, new function and new product).
  - The problem solving loop applies to software engineering work at different levels.
  - In Figure , each stage in the problem solving loop contain as identical problem solving loop, which contains still another problem solving loop.


- 
- In reality, it is difficult to separate activities clearly because cross talk within different stages.
  - The simple view leads to an important conclusion – regardless of the process model, all of stages- status quo, problem definition, technical development and solution integration – coexist simultaneously at some level detail.
  - All four stages apply to the analysis of a complete application and to the generation of a small segment of code.

- 
- As work progresses toward complete system, the stages are applied recursively to user's needs and the development's technical specification of the software.
  - Following process models for software engineering represents activities to properly design the software.

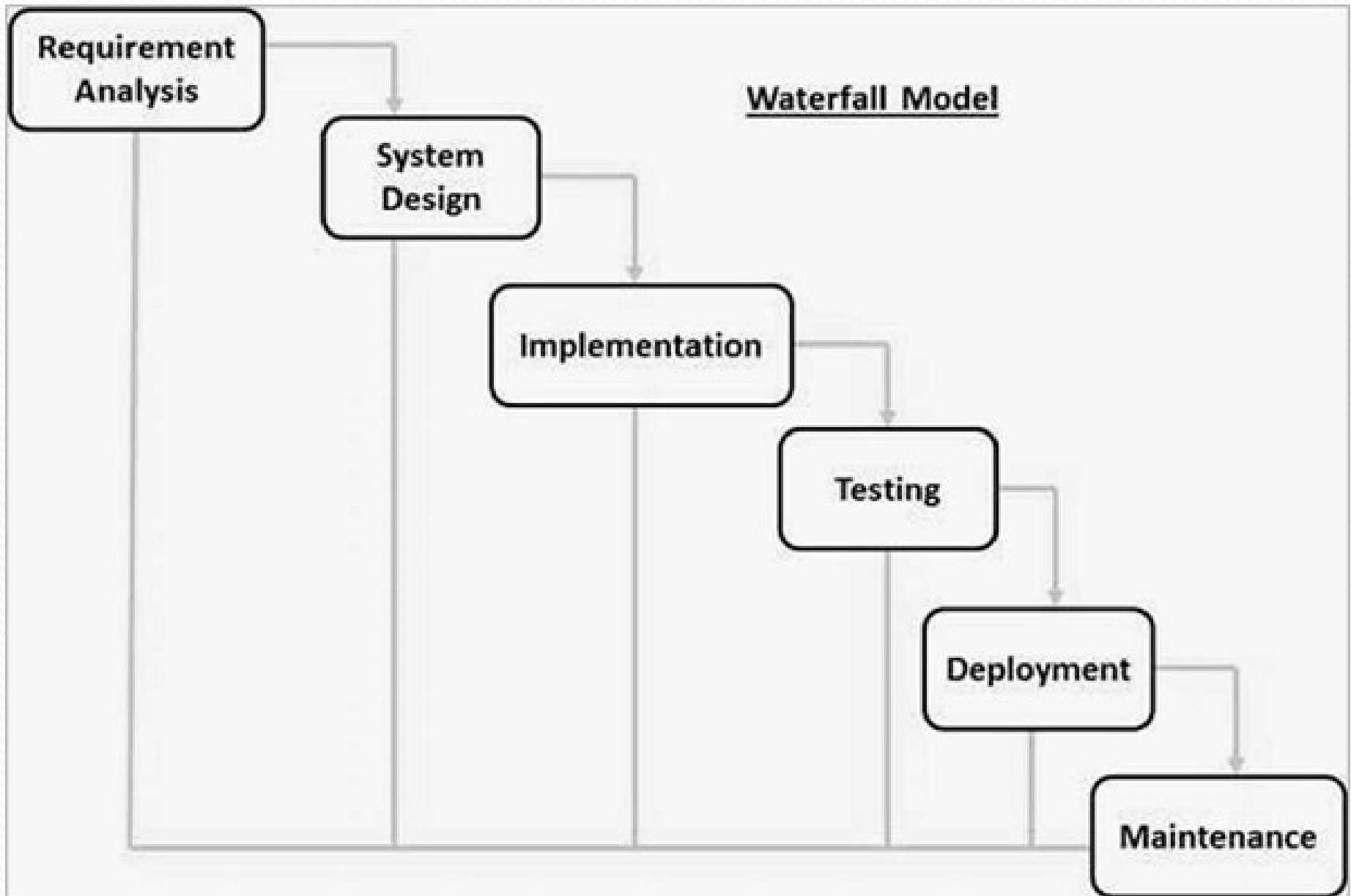


# SDLC - Waterfall Model

- The Waterfall Model was the first Process Model to be introduced. It is also referred to as a **linear-sequential life cycle model**.
- It is very simple to understand and use. In a waterfall model, each phase must be completed before the next phase can begin and there is no overlapping in the phases.
- The Waterfall model is the earliest SDLC approach that was used for software development.

- 
- Waterfall approach was first SDLC Model to be used widely in Software Engineering to ensure success of the project.
  - In "The Waterfall" approach, the whole process of software development is divided into separate phases.
  - In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially.
  - The following illustration is a representation of the different phases of the Waterfall Model.

## Waterfall Model






The sequential phases in Waterfall model are –

- **Requirement Gathering and analysis** – All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.
- **System Design** – The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
- **Implementation** – With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.

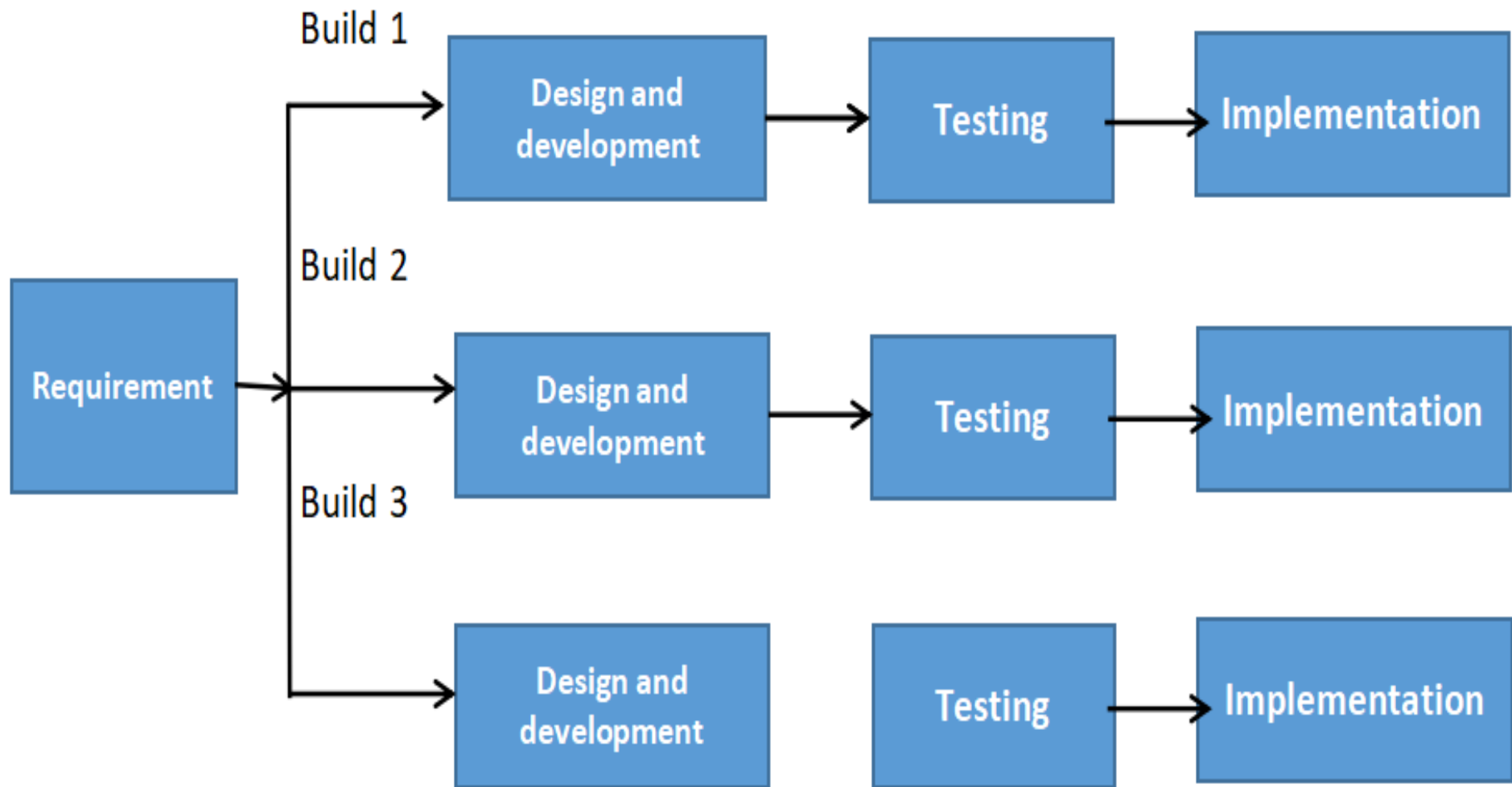



- 
- **Integration and Testing** – All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
  - **Deployment of system** – Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
  - **Maintenance** – There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.



# SDLC - Iterative Model

- Iterative process starts with a simple implementation of a subset of the software requirements and iteratively enhances the evolving versions until the full system is implemented.
- At each iteration, design modifications are made and new functional capabilities are added.
- The basic idea behind this method is to develop a system through repeated cycles (iterative) and in smaller portions at a time (incremental).
- The following illustration is a representation of the Iterative and Incremental model –



- 
- Iterative and Incremental development is a combination of both iterative design or iterative method and incremental build model for development.

### **Advantages**

- The advantage of this model is that there is a working model of the system at a very early stage of development, which makes it easier to find functional or design flaws.
- Finding issues at an early stage of development enables to take corrective measures in a limited budget.



## Disadvantages

- The disadvantage with this SDLC model is that it is applicable only to large and bulky software development projects.
- This is because it is hard to break a small software system into further small serviceable increments/modules.




# V-Model(Verification and Validation)

- The V-model is a type of SDLC model where process executes in a sequential manner in V-shape.
- It is also known as Verification and Validation model. It is based on the association of a testing phase for each corresponding development stage.
- Development of each step directly associated with the testing phase.
- The next phase starts only after completion of the previous phase i.e. for each development activity, there is a testing activity corresponding to it.





## Validation Phases

- 
- **Verification:** It involves static analysis technique (review) done without executing code.
  - It is the process of evaluation of the product development phase to find whether specified requirements meet.
  - **Validation:** It involves dynamic analysis technique (functional, non-functional), testing done by executing code.
  - Validation is the process to evaluate the software after the completion of the development phase to determine whether software meets the customer expectations and requirements.
  - So V-Model contains Verification phases on one side of the Validation phases on the other side. Verification and Validation phases are joined by coding phase in V-shape. Thus it is called V-Model.

## Design Phase:(Verification Phases)

- **Requirement Analysis:** This phase contains detailed communication with the customer to understand their requirements and expectations. This stage is known as Requirement Gathering.
- **System Design:** This phase contains the system design and the complete hardware and communication setup for developing product.
- **Architectural Design:** System design is broken down further into modules taking up different functionalities. The data transfer and communication between the internal modules and with the outside world (other systems) is clearly understood.
- **Module Design:** In this phase the system breaks down into small modules. The detailed design of modules is specified, also known as Low-Level Design (LLD).

## Testing Phases:(Validation Phases)

- **Unit Testing:** Unit Test Plans are developed during module design phase. These Unit Test Plans are executed to eliminate bugs at code or unit level.
- **Integration testing:** After completion of unit testing Integration testing is performed. In integration testing, the modules are integrated and the system is tested. Integration testing is performed on the Architecture design phase.
- **System Testing:** System testing test the complete application with its functionality, inter dependency, and communication.It tests the functional and non-functional requirements of the developed application.
- **User Acceptance Testing (UAT):** UAT is performed in a user environment that resembles the production environment. UAT verifies that the delivered system meets user's requirement and system is ready for use in real world.



## **Advantages:**

- 
- 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.

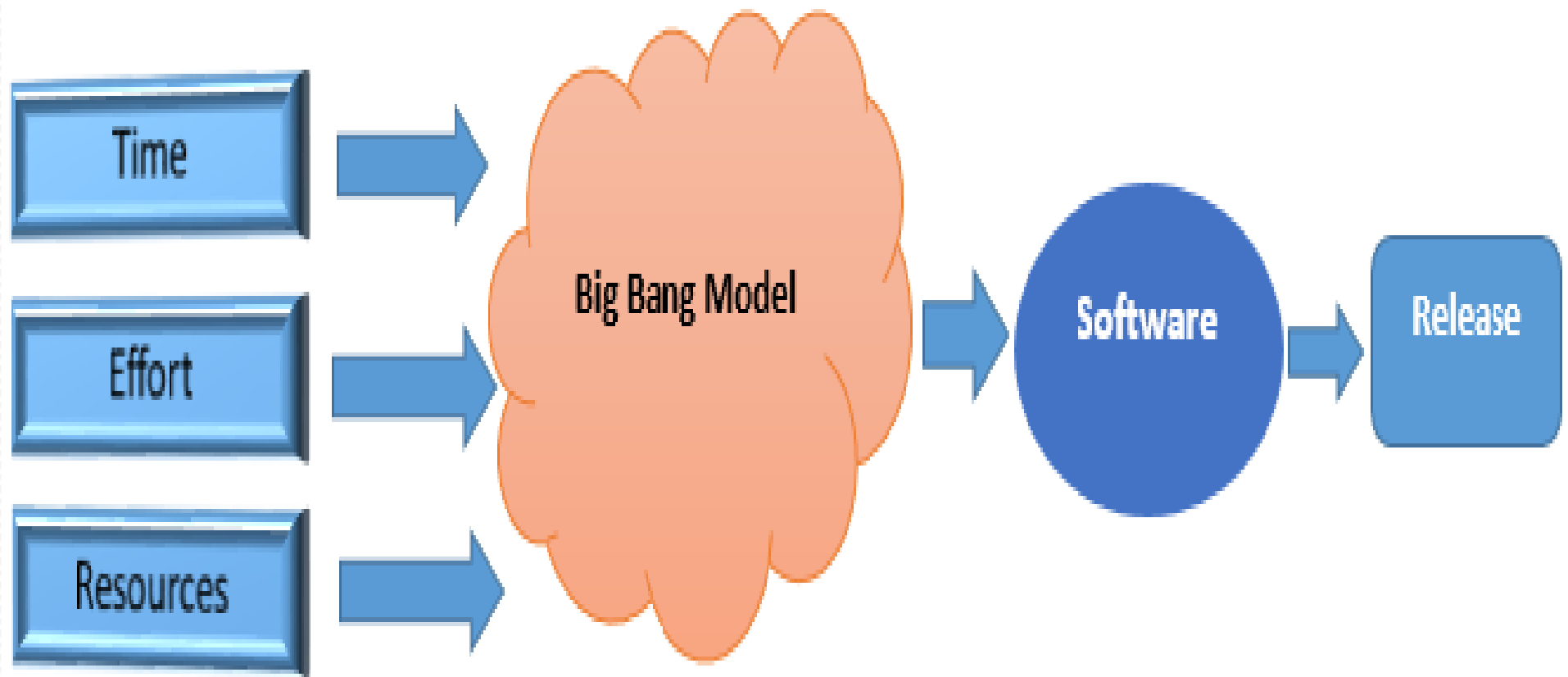
## **Disadvantages:**

- 
- High risk and It does not easily handle concurrent events.
- It is not a good for complex and object-oriented projects.
- This model does not support iteration of phases.



# Big Bang Model

- The Big bang model is an SDLC model that starts from nothing.
- It is the simplest model in SDLC (Software Development Life Cycle) as it requires almost no planning. However, it requires lots of funds and coding and takes more time.
- The name big bang model was set after the “Great Big Bang” which led to the development of galaxies, stars, planets, etc. Similarly, this SDLC model combines time, efforts, and resources to build a product.
- The product is gradually built as the requirements from the customer come, however, the end product might not meet the actual requirements.



**Big Bang SDLC MODEL**

# Features of Big Bang Model :

- Not require a well-documented requirement specification.
- Provides a quick overview of the prototype.
- Needs little effort and the idea of implementation.
- Allows merging of newer technologies to see the changes and adaptability.

## Advantages:

- There is no planning required for this.
- Suitable for small projects.
- Very few resources are required.
- Easy to implement.
- Very much flexible for the developers working on it.

## Disadvantages:

- Not suitable for large projects.
- Highly risky model and uncertain.
- Might be expensive if requirements are not clear

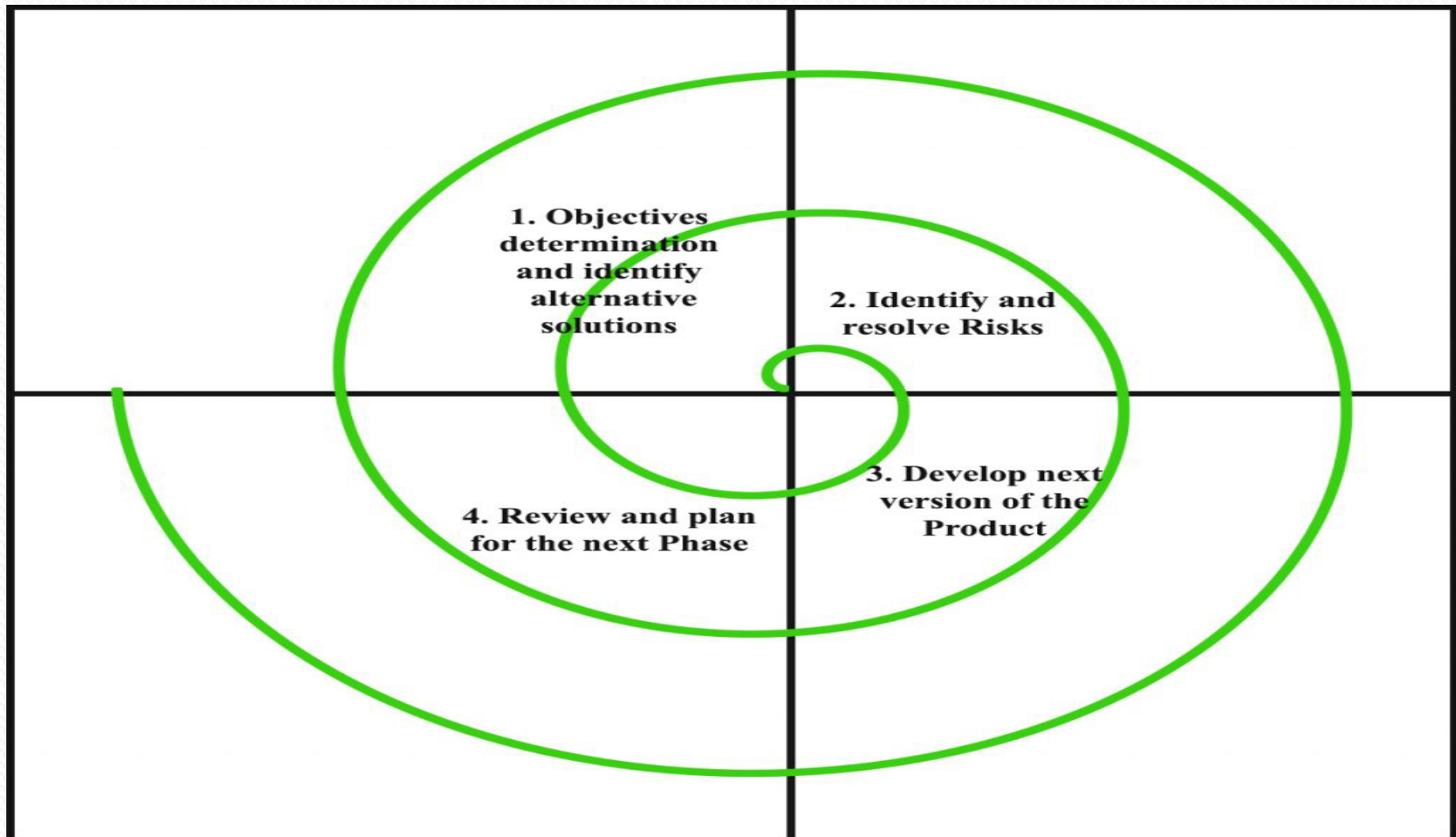



# SDLC - Spiral Model

- The spiral model is a **systems development lifecycle (SDLC) method used for risk management that combines the iterative development process model with elements of the Waterfall model.**
- The spiral model is used by software engineers and is favored for large, expensive and complicated projects.
- **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.
- Each loop of the spiral is called a **Phase of the software development process**. 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, so the project manager has an important role to develop a product using the spiral model.



- 
- Each phase of the Spiral Model is divided into four quadrants as shown in the above figure.

### **1. Objectives determination and identify alternative solutions:**

Requirements are gathered from the customers and the objectives are identified, elaborated, and analyzed at the start of every phase.

Then alternative solutions possible for the phase are proposed in this quadrant.

### **2. Identify and resolve Risks:** 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 and the risks are resolved using the best possible strategy. At the end of this quadrant, the Prototype is built for the best possible solution.



### **3. Develop 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.

## Advantages:

- Software is produced early in the software life cycle.
- Risk handling is one of important advantages of the Spiral model, it is best development model to follow due to the risk analysis and risk handling at every phase.
- It is good for large and complex projects.
- It is good for customer satisfaction. We can involve customers in the development of products at early phase of the software development.
- Strong approval and documentation control.
- It is suitable for high risk projects, where business needs may be unstable. A highly customized product can be developed using this.



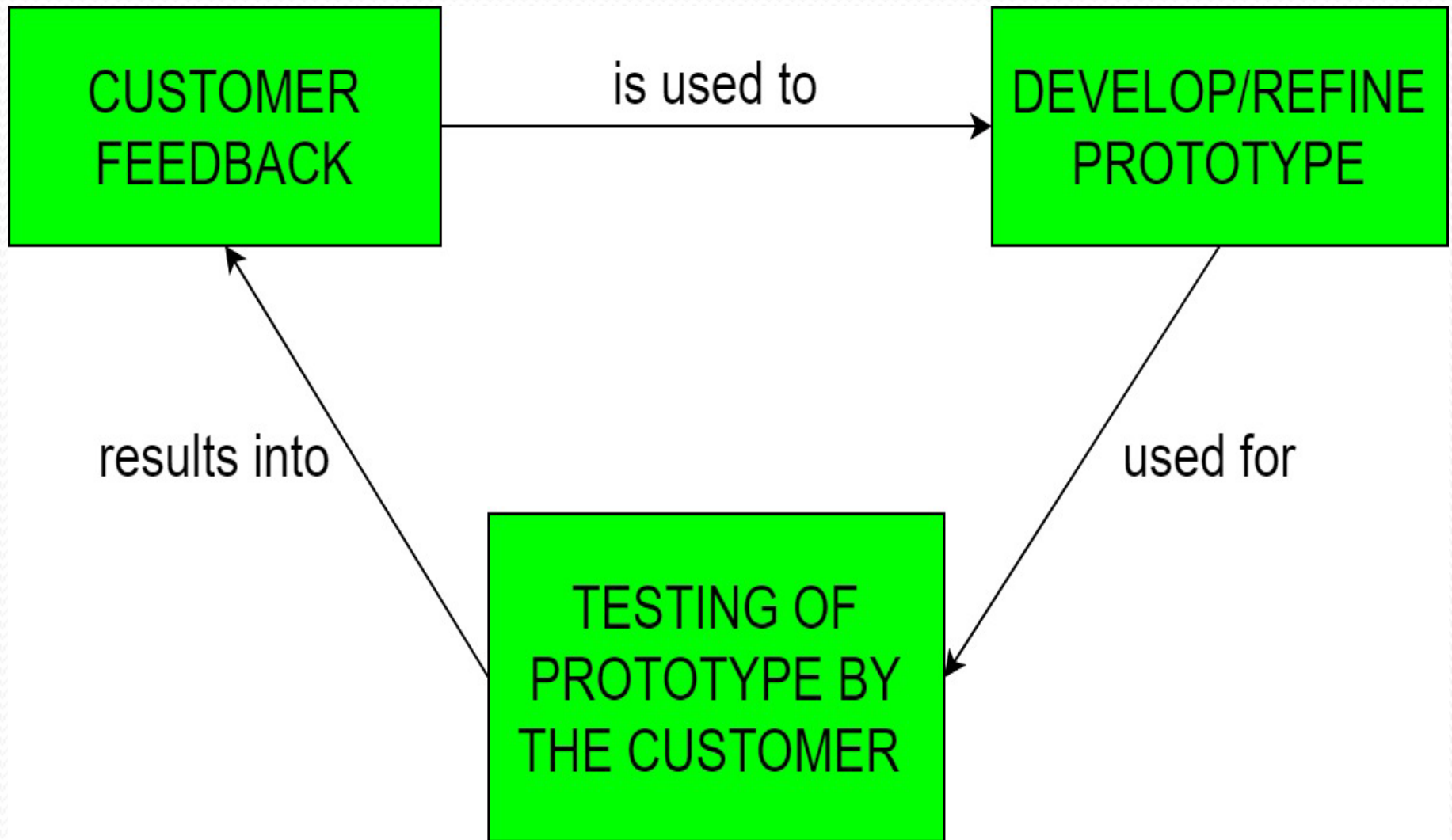


## **Disadvantages:**

- It is not suitable for small projects as it is expensive.
- It is much more complex than other SDLC models. Process is complex.
- Difficulty in time management.
- It is not suitable for low risk projects.

# SDLC-Prototyping Model

- The Prototyping Model is one of the most popularly used Software Development Life Cycle Models (SDLC models).
- Prototyping is defined as the process of developing a working replication of a product or system that has to be engineered.
- 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.



## SDLC-Prototyping Model



There are four types of models available:

**Rapid Throwaway Prototyping –**

This technique offers a useful method of exploring ideas and getting customer feedback for each of them.

**Evolutionary Prototyping –**

In this method, the prototype developed initially is incrementally refined on the basis of customer feedback till it finally gets accepted.

**Incremental Prototyping –** In this type of incremental Prototyping, the final expected product is broken into different small pieces of prototypes and being developed individually.

**Extreme Prototyping-** This method is mainly used for web development.





## Advantages:

- This model is flexible in design.
- It is easy to detect errors.
- We can find missing functionality easily.
- There is scope of refinement, it means new requirements can be easily Implementation.
- It is ideal for online system.



## **Disadvantages:**

- This model is costly.
- It has poor documentation because of continuously changing customer requirements.
- Customers sometimes demand the actual product to be delivered soon after seeing an early prototype.