# SEQA
# Session 1

Software Development Life Cycle (SDLC) Models

Dr. Bharati Wukkadada

# contents

- Waterfall Model,
- Prototype Model,
- Spiral Model,
- Evolutionary Development Models-
- Iterative model and incremental model

# question

- When you work to build a product / system, it's important to go through a series of predictable steps- a roadmap that helps you create a timely, high-quality result. The roadmap that you follow is called a " ....."

# Answer is

Software Process

# software **process .**

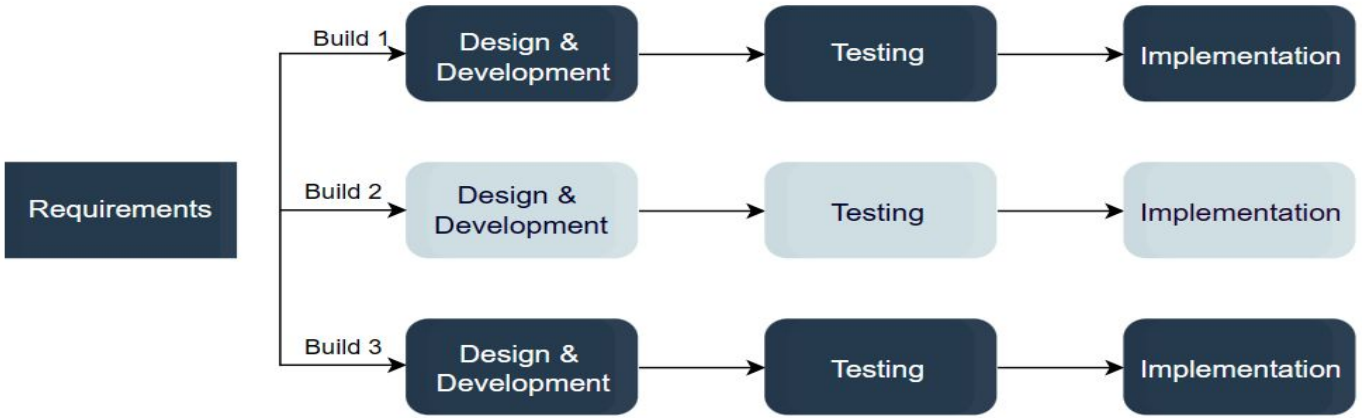
# Process Flow

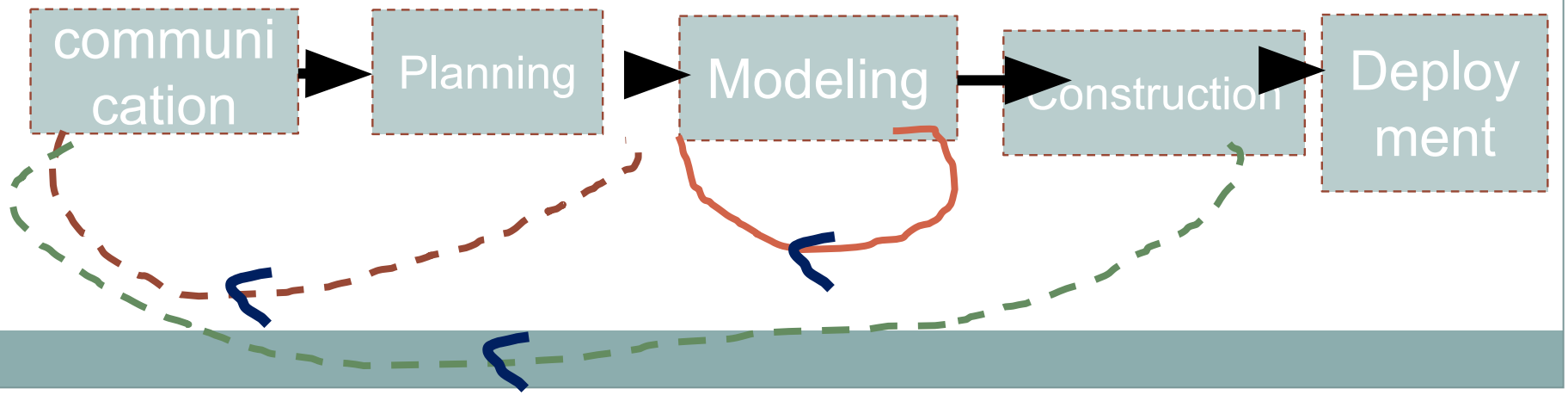
## a. Linear Process Flow

## CPMCD

# b. Iterative process flow

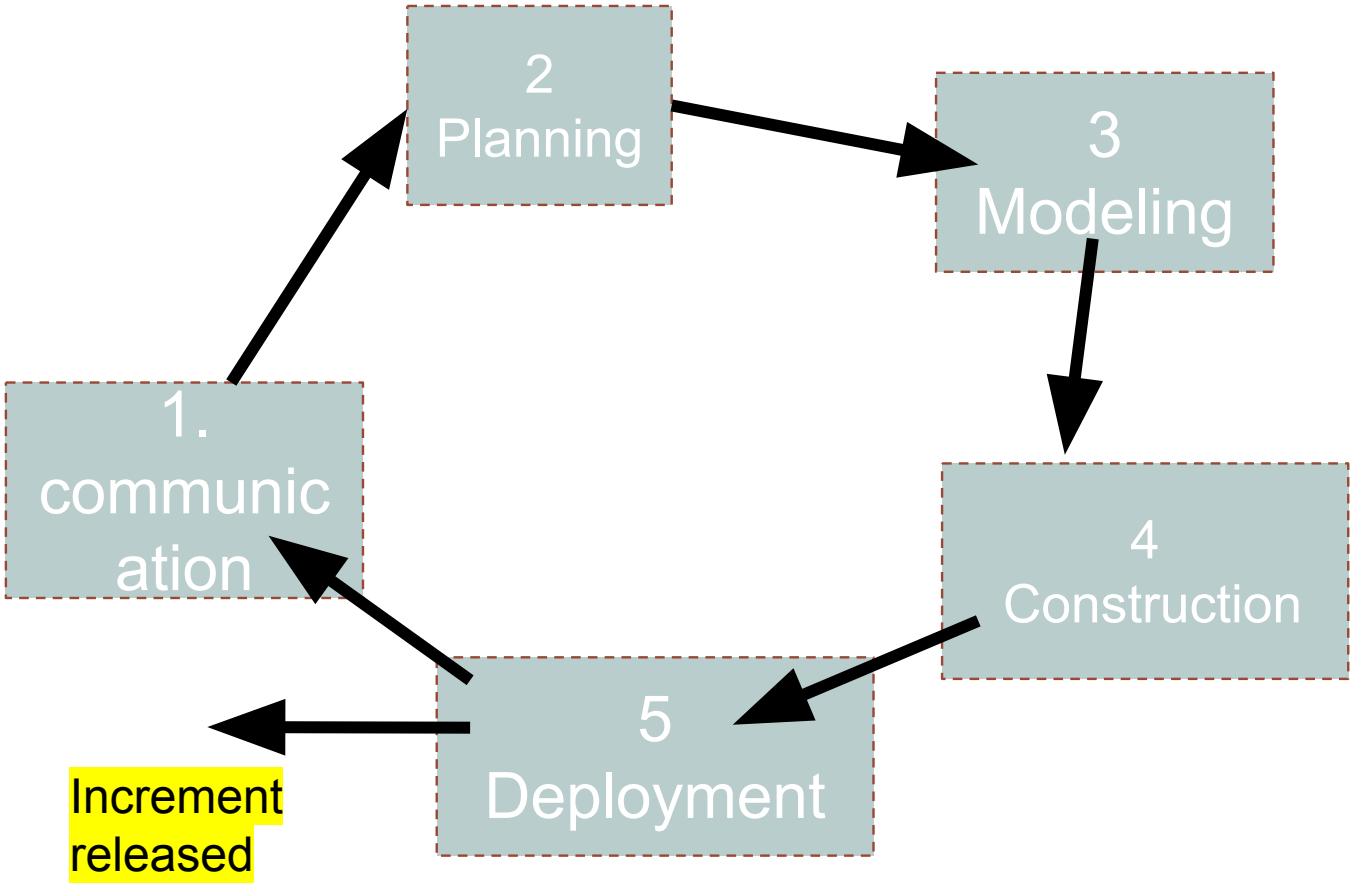
CPMCD



# c. Evolutionary Process Flow

CPMCD

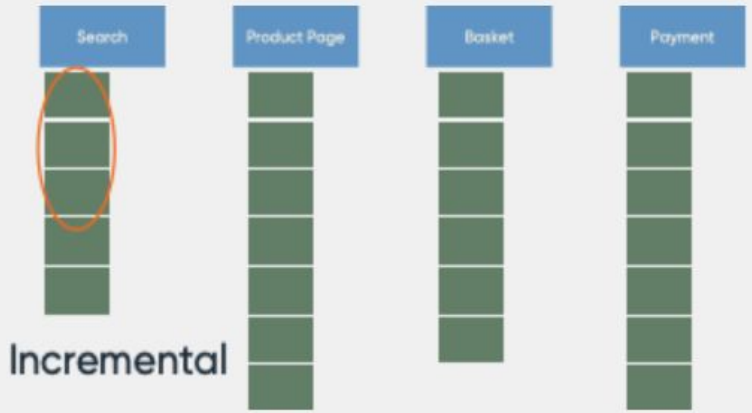


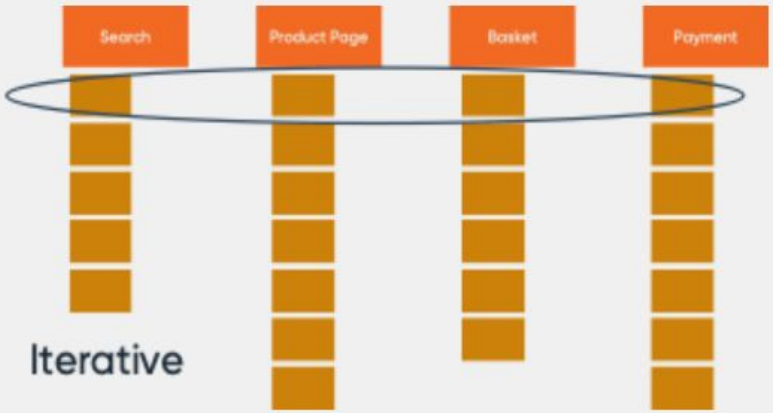
**Incremental** Incremental development is a development approach that slices the product into fully working slices that are called increments.

**Iterative** development is when teams gradually build up the features and functions but don't wait until each of these is complete before releasing.

Vision is fully clear(small and progressive)

**Vision is not fully defined(repitative)**

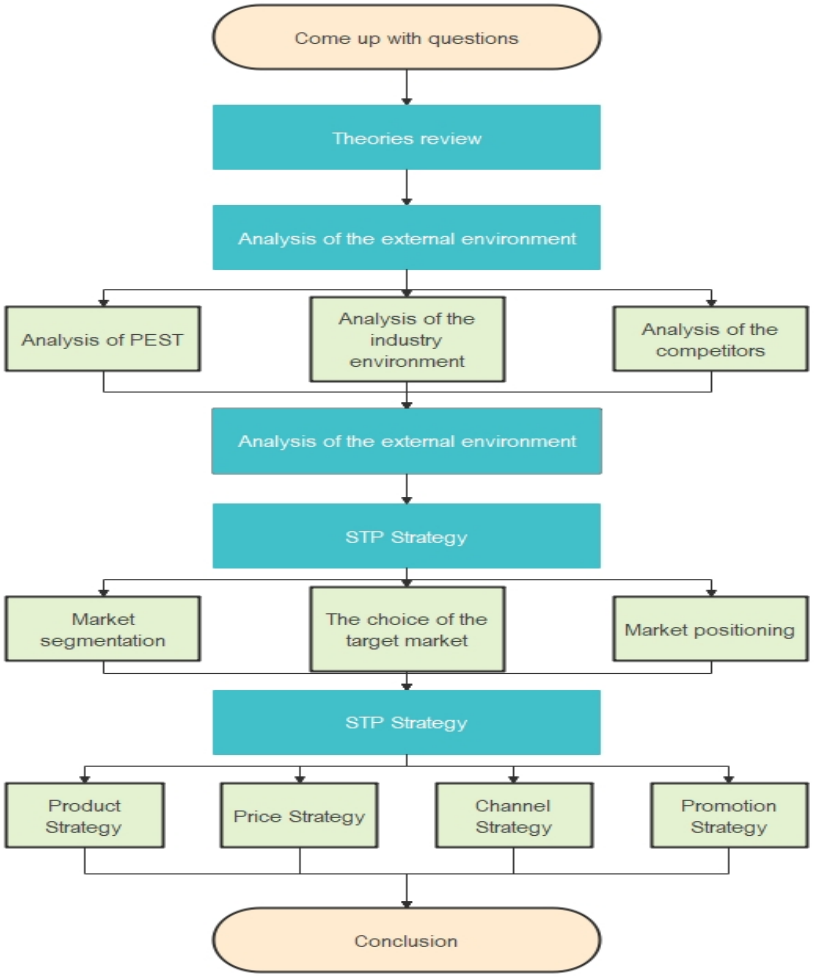



- **Incremental development** emphasizes building software in distinct stages, with each stage adding new features or functionality to the final product.
- **Iterative development** focuses on continuous improvement through repeated cycles.

- In summary:
  - **Iterative development** focuses on continuous improvement through repeated cycles.
  - **Incremental development** emphasizes building software in distinct stages, with each stage adding new features or functionality to the final product.
  - **Iterative** is ideal when the vision isn't fully defined, while **incremental** works well when the vision is clear and you want feedback during refinement

# d. parallel process flow



# Software lifecycle

- According to IEEE standard software life cycle is
- " the period of time that starts when a s/w product is started and ends when the product is no longer available for use . The s/w life cycle typically includes a requirement , design, implementation, test , installation and checkout, operation and maintenance phase, and sometimes retirement phase

A not-for-profit organization, **Institute of Electrical and Electronics Engineers** (**IEEE**) is the world's largest technical professional organization dedicated to advancing technology for the benefit of humanity.

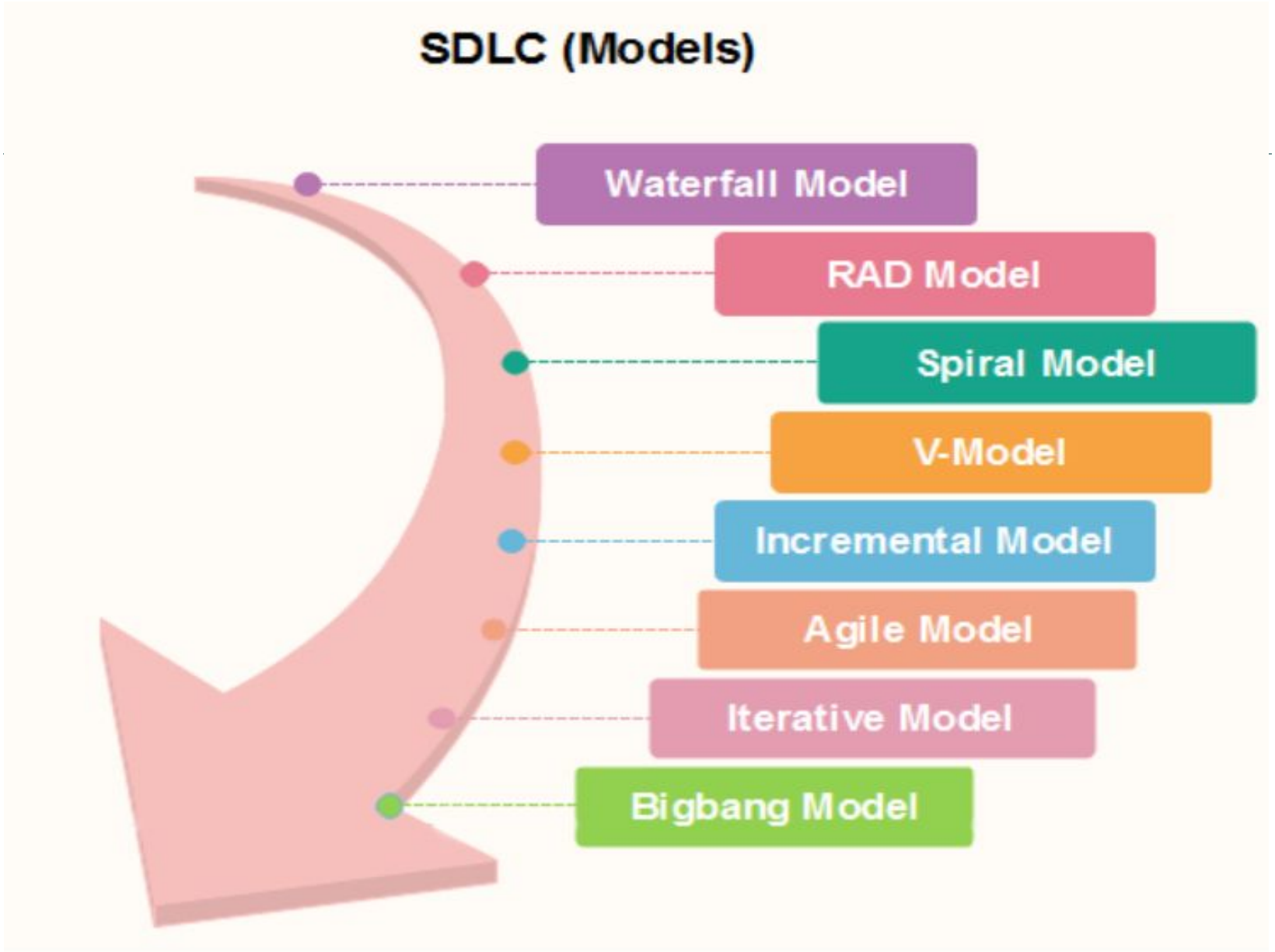
# SDLC MODEL

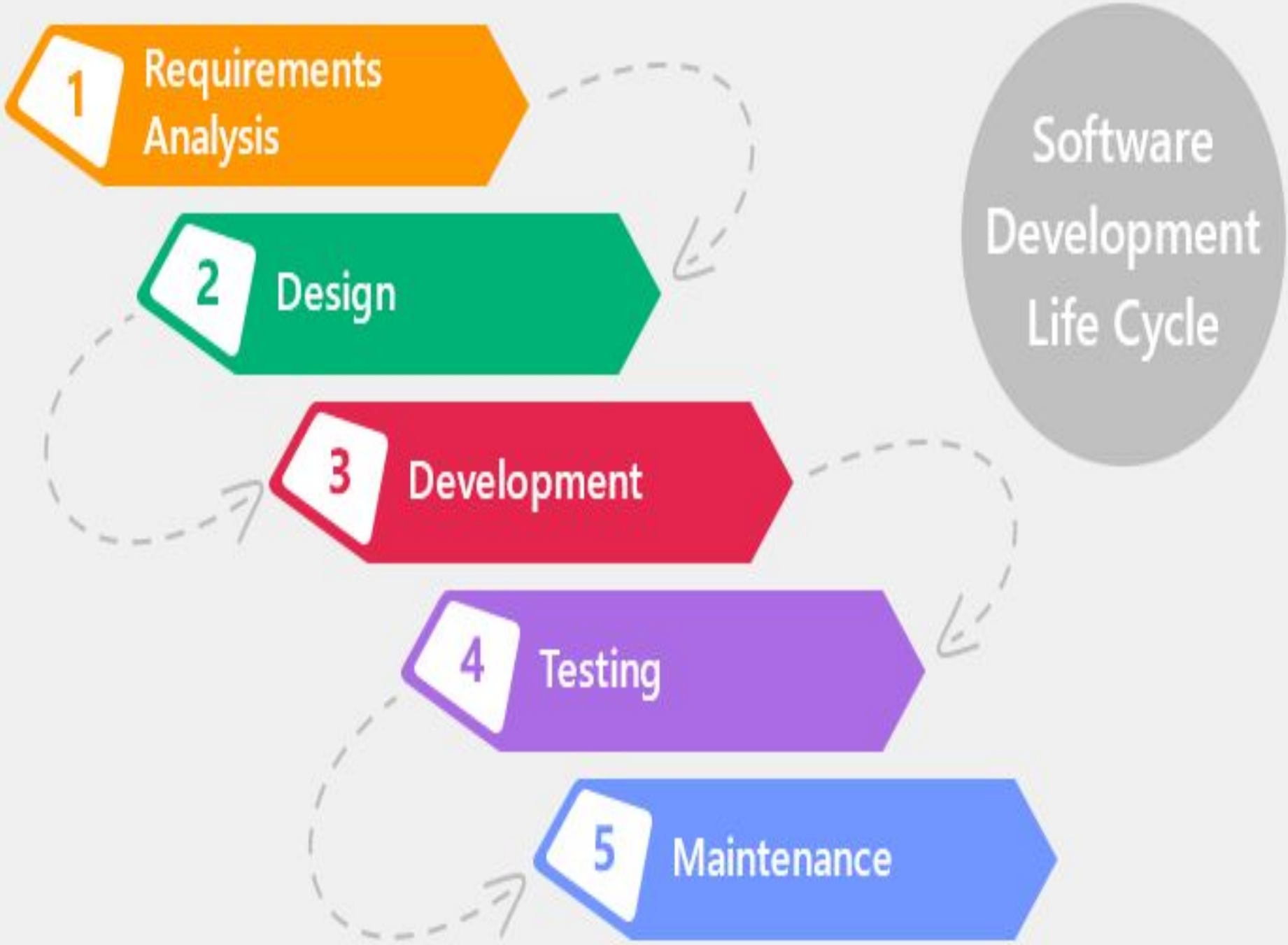
- Build and fix model
- Waterfall model
- Prototype model
- Spiral model
- Evolutionary development models-
- Iterative model
- Incremental model

- Waterfall Model,
- Prototype Model,
- Spiral Model,
- Evolutionary Development Models-
- Iterative model and incremental model

- Software Development life cycle (SDLC) is a spiritual model used in project management that defines the stages include in an information system development project, from an initial feasibility study to the maintenance of the completed application.

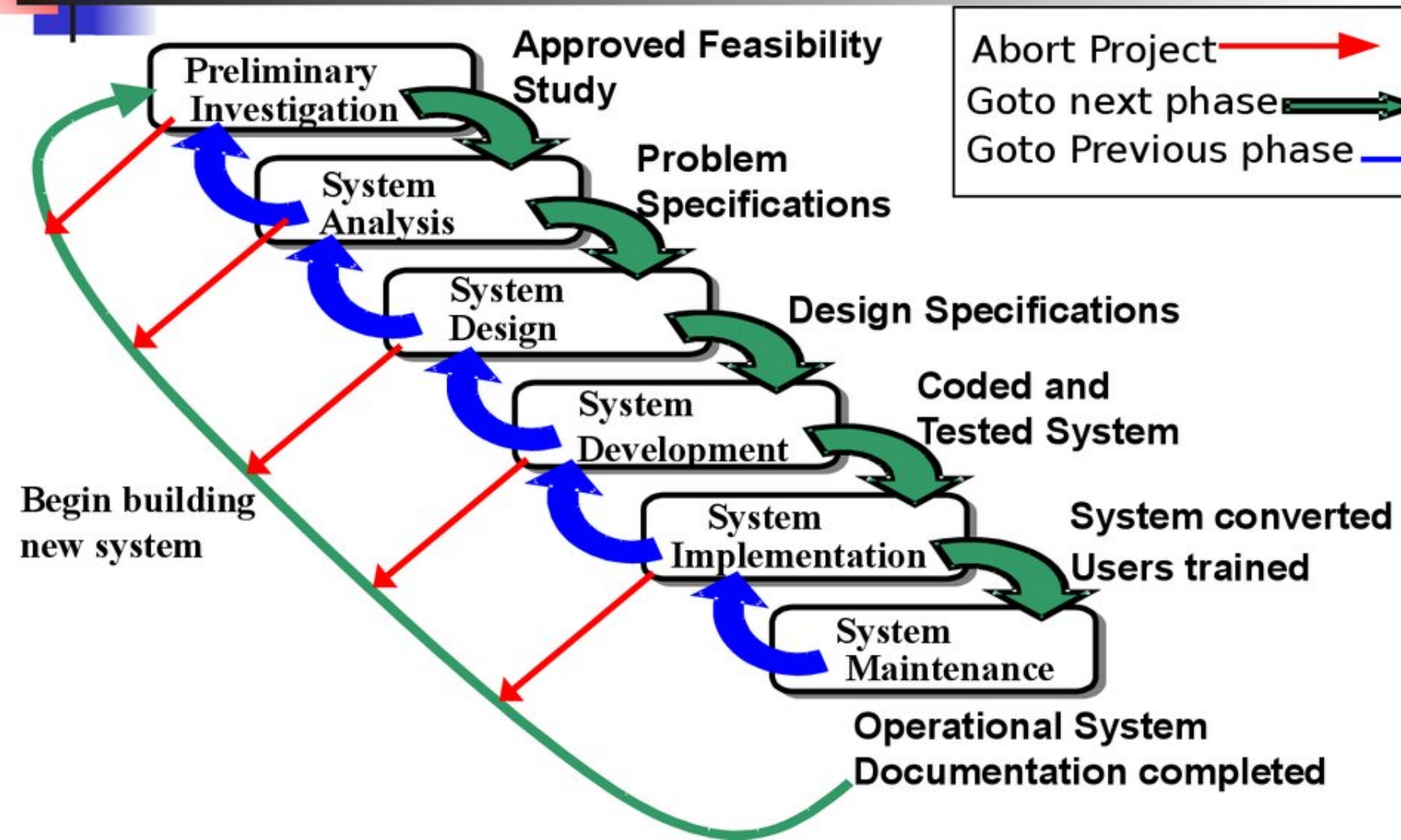
- There are different software development life cycle models specify and design, which are followed during the software development phase. These models are also called "**Software Development Process Models**." Each process model follows a series of phase unique to its type to ensure success in the step of software development.


SDLC (Models)
Waterfall Model
RAD Model
Spiral Model
V-Model
Incremental Model
Agile Model
Iterative Model
Bigbang Model


1
Requirements Analysis
2
Design
3
Development
4
Testing
5
Maintenance
Software Development Life Cycle

# Software Development Life Cycle

# Deliverables of the SDLC

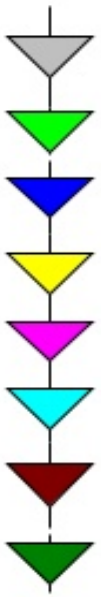


# SDLC - Requirement Gathering

## What Are the 8 Characteristics of Good User Requirements?

**A user requirement is good if it is:**

1. Verifiable
2. Clear and concise
3. Complete
4. Consistent
5. Traceable
6. Viable
7. Necessary
8. Implementation free



**Think of these characteristics as a series of filters. A good requirement will pass through all eight filters.**

viable is capable of doing what it is intended to do

# SDLC - Design

## Why design is important?

A good design is the key to successful product. Few desirable characteristics that every good software design must possess are:

1. Correctness
2. Understandability
3. Efficiency
4. Maintainability

Without well-design system, we risk building an unstable system.

- One that will fail when small changes are made.
- One that will be difficult to maintain.

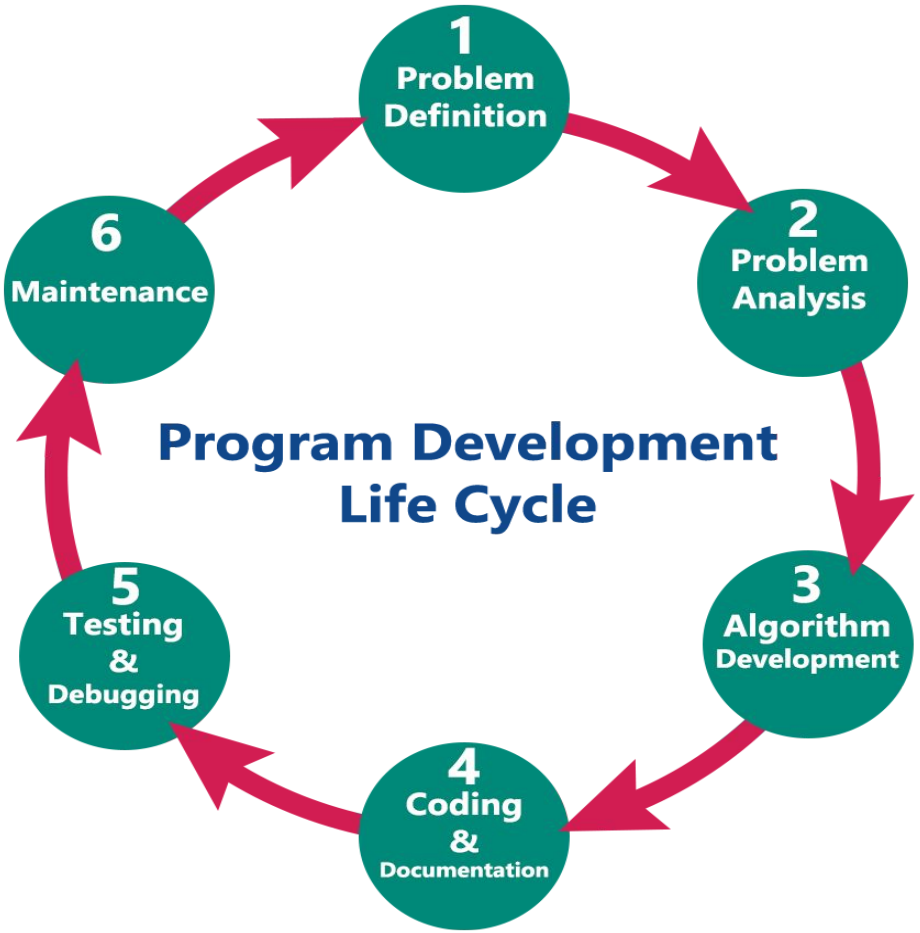
Maintainability:

The ability of an item, under stated conditions of use, to be retained in

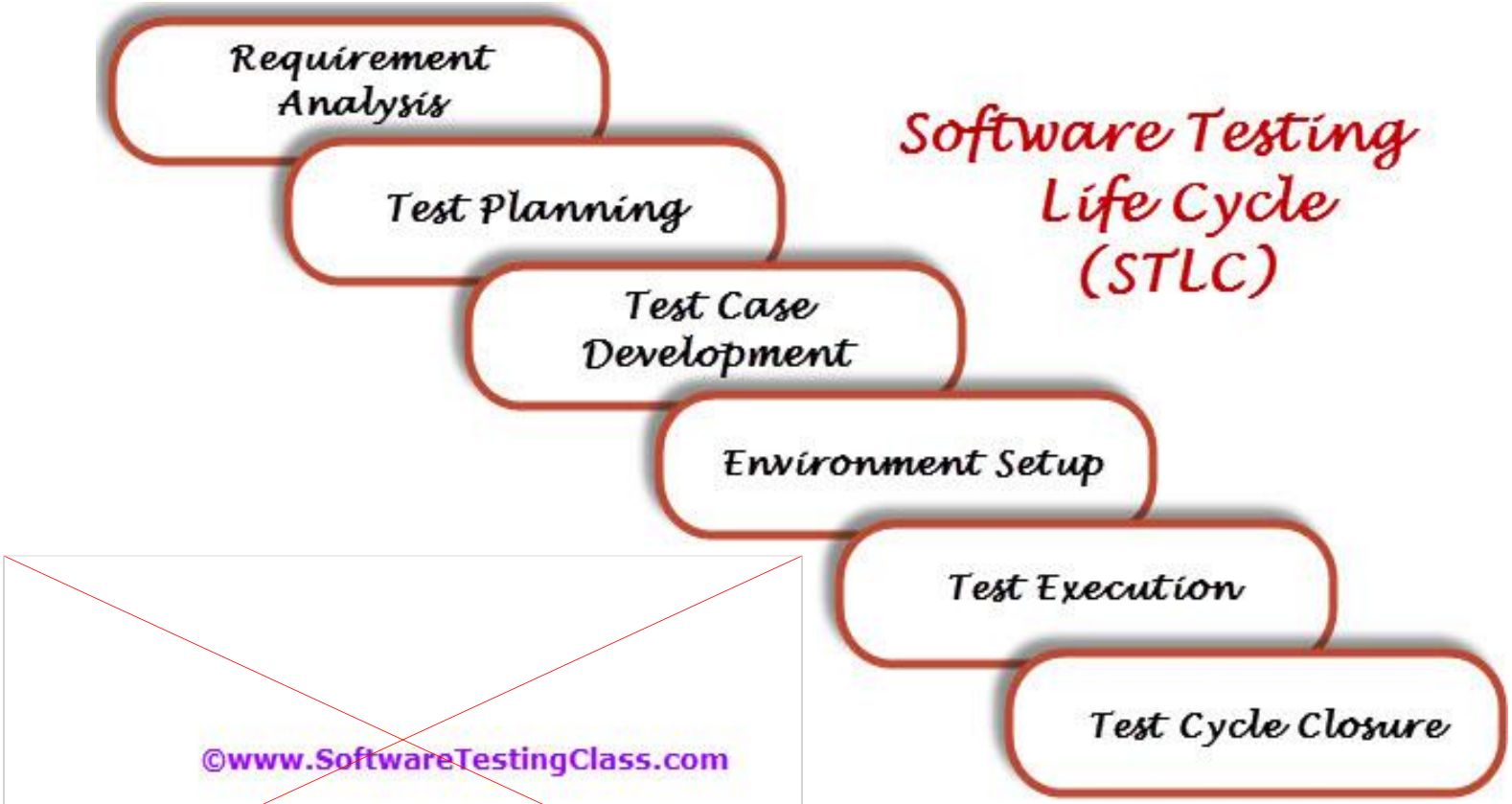
# SDLC – Design

## example for web design life cycle

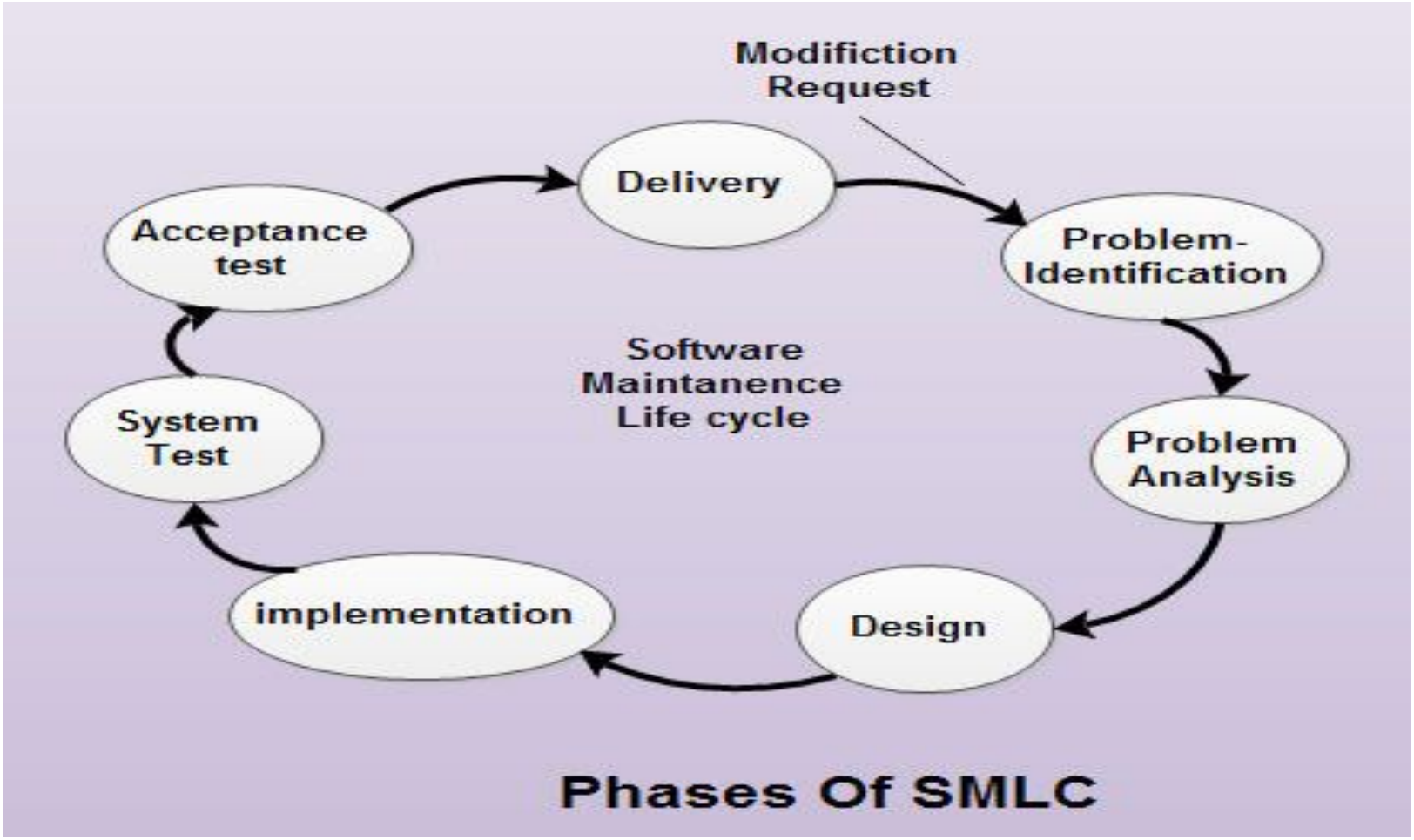
# SDLC – Coding/ Develop



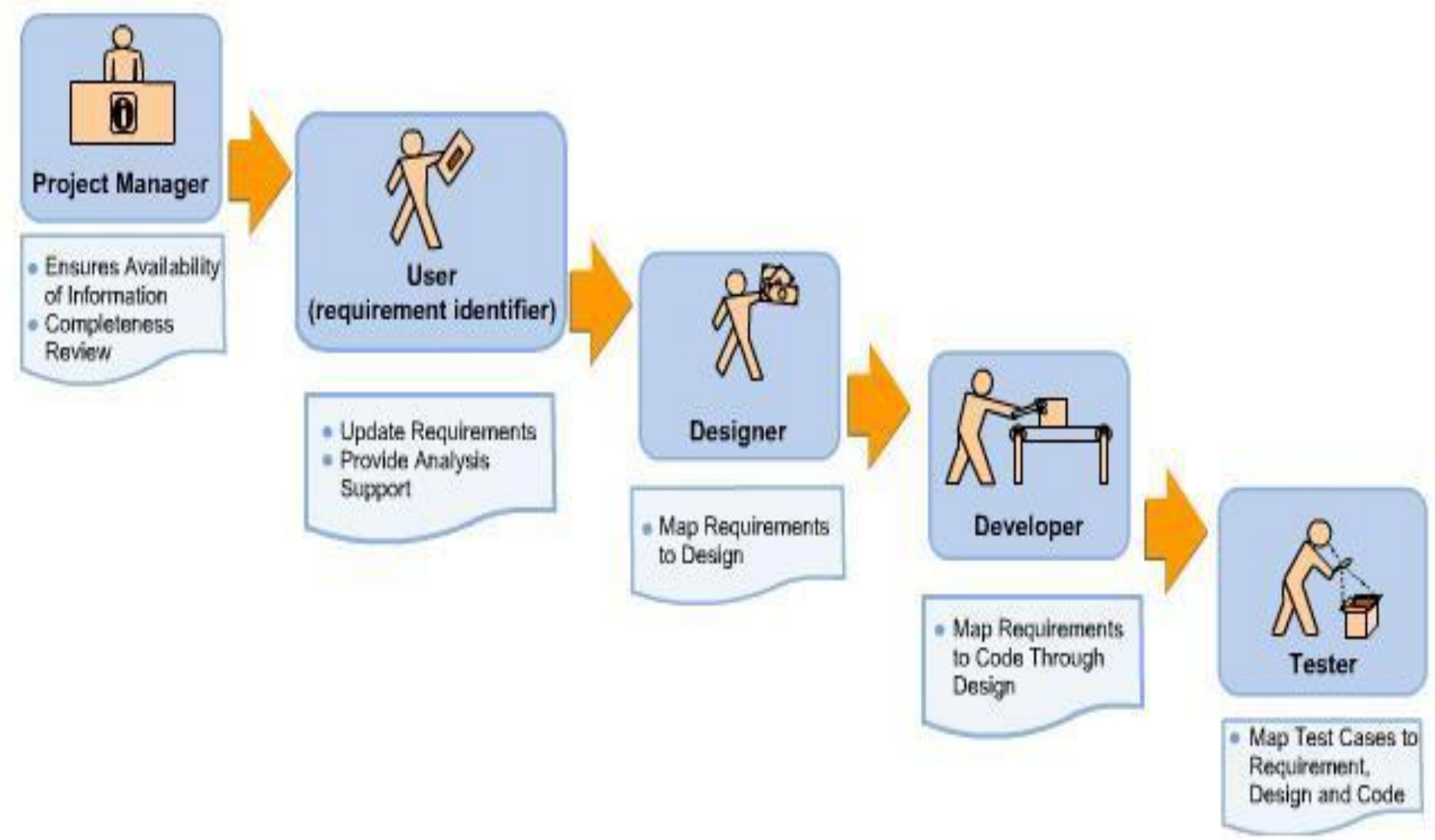
# SDLC – Testing



# SDLC - **Maintenance**



# Roles & Responsibilities



# The role of project manager

# The role of project manager


2

1

# The Role of the Project Manager

- Project manager coordinates project development
- Specifications in a detailed plan at project inception
  - Activities that must take place
  - The deliverables that must be produced
  - Resources needed
- Project manager accountable for success or failure
- Project manager has internal/external responsibilities
- Many career paths lead to project management

# Roles and Responsibilities

- Risk Analysis
- Managing Risks and Issues
- Monitoring and Reporting Progress
- Team Leadership
- Strategic Influencing
- Business Partnering
- Working with Vendors
- Scalability, Interoperability and Portability Analysis
- Controlling Quality
- Benefits Realization

**Benefits Realisation:** planning, structuring and actual **realisation** of the **benefits** of a business change or business improvement project. Despite vast sums being spent on projects

## **Analysis** for **Interoper ability**, **Portability** an d **Scalability**

To what degree can the software be transferred from one environment to another? To what degree can a network be enhanced without a major (fundamental) change in design?

**Scalability:** It is the ability of a computer application or product (hardware or software) to continue to function well as it (or its context) is changed in size or volume

**interoperable** : if two products, programs, etc. are interoperable, they can be used together:

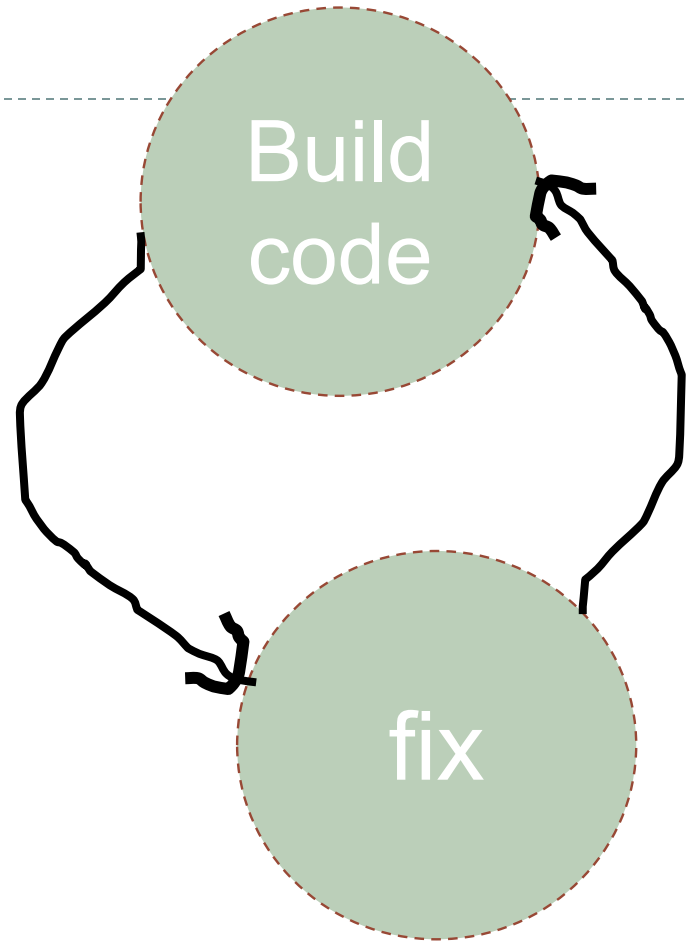
**Portability** in high-level computer programming is the usability of the same **software** in different environments.

# BUILD AND FIX MODEL

# Build and Fix Model

- Sometimes a product is constructed without specifications or any attempt at design
- Developer simply builds, reworked as many times as necessary to satisfy the client
- It is adhoc approach and not well defined
- Simple two phase model- write code, error correction
- It works well on small programming exercise 100-200 lines long

- It is totally unsatisfactory for s/w of any reasonable size. Code soon becomes unfixable and un-enhanceable
- The cost of the development using this approach is actually very high as compared to the other specified and designed product
- Maintenance of the product can be extremely difficulty without specifications/design documents



- Waterfall Model,
- Prototype Model,
- Spiral Model,
- Evolutionary Development Models-
- Iterative model and incremental model

# waterfall model

***a project management methodology based on a sequential design process***

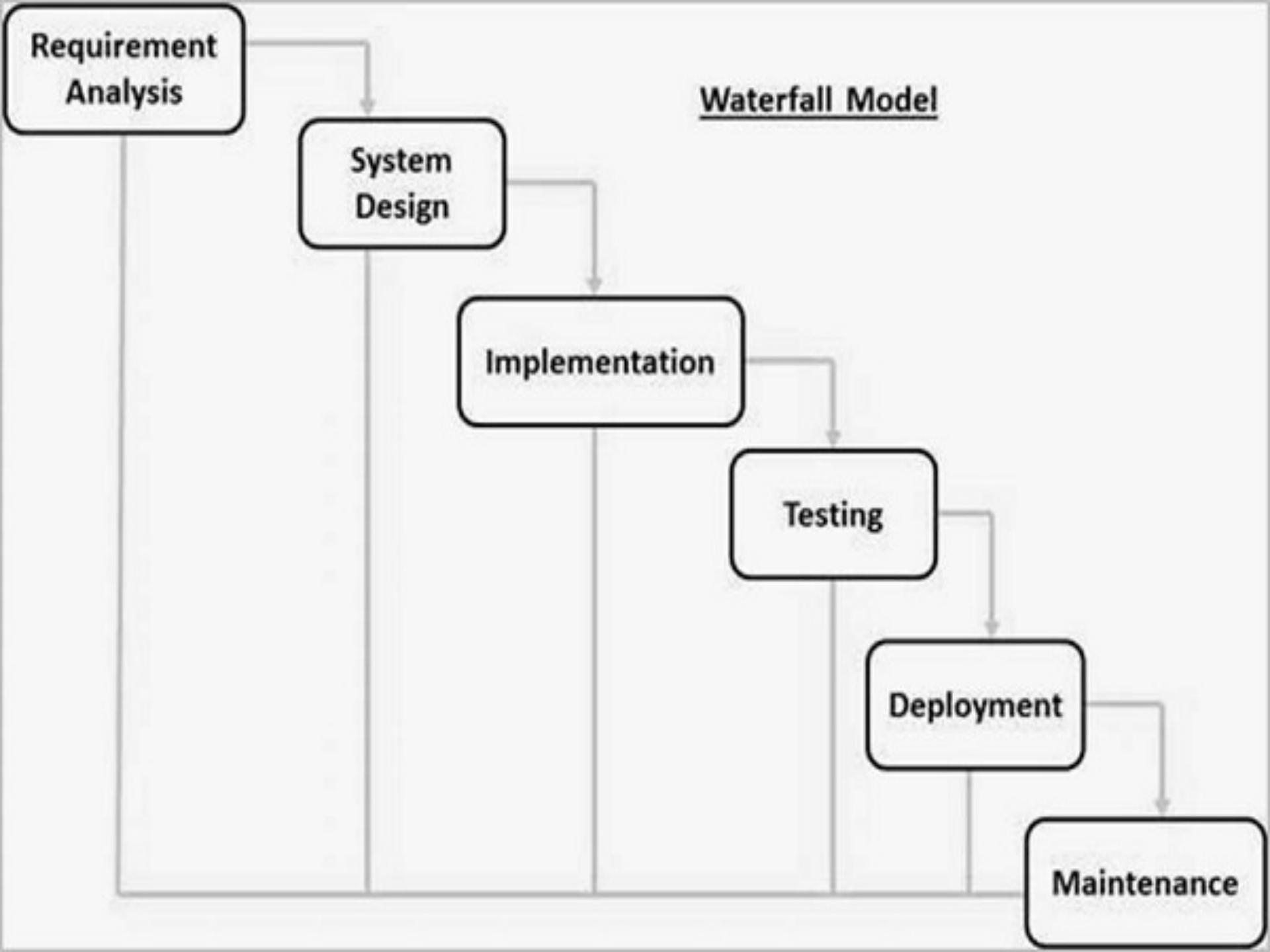
# Waterfall Model

- This model is also known as the **linear sequential model** or **classic Life cycle Model**.

- **demands a systematic and sequential approach to software development** that **begins at the system level**

- The waterfall is a universally accepted SDLC model. In this method, the whole process of software development is divided into various phases.
- The waterfall model is a continuous software development model in which development is seen as flowing steadily downwards (like a waterfall) through the steps of requirements analysis, design, implementation, testing (validation), integration, and maintenance.

- Linear ordering of activities has some significant consequences. First, to identify the end of a phase and the beginning of the next, some certification techniques have to be employed at the end of each step. Some verification and validation usually do this mean that will ensure that the output of the stage is consistent with its input (which is the output of the previous step), and that the output of the stage is consistent with the overall requirements of the system.


Requirement Analysis
System Design
Implementation
Testing
Deployment
Maintenance
Waterfall Model

## 1. Requirement Gathering and analysis –

- **All possible requirements of the system to be developed are captured** in this phase
  - understand exact requirements,
  - customer goals,
  - the requirements describe “what “ of a system, not the “how”

- **documented in a requirement specification document**.(srs)

If the developer fails to implement full set of requirements , it may amount to failure to implement the constructed system

## 2. System Design –

**The requirement specifications from first phase are studied** in this phase

and

**the system design is prepared**. [SDD s/w design description]

This **system design helps in:**

- **specifying hardware and system requirements**
- **defining the overall system architecture**
- **SDD is sufficient to begin coding phase**

## 3. 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**.

**Small modules are tested in isolation from the rest of the s/w product**

## 4. 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**.

## 5. 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.**

## 6. Maintenance –

- s/w maintenance is a task that every development group has to face
- 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**.

- This model is easy to understand and reinforces the notation of

  "define before design" and "design before code"
- This model expects complete and accurate requirements early in the process, delay will be serious errors
- It does not incorporates any kind of risk assessment

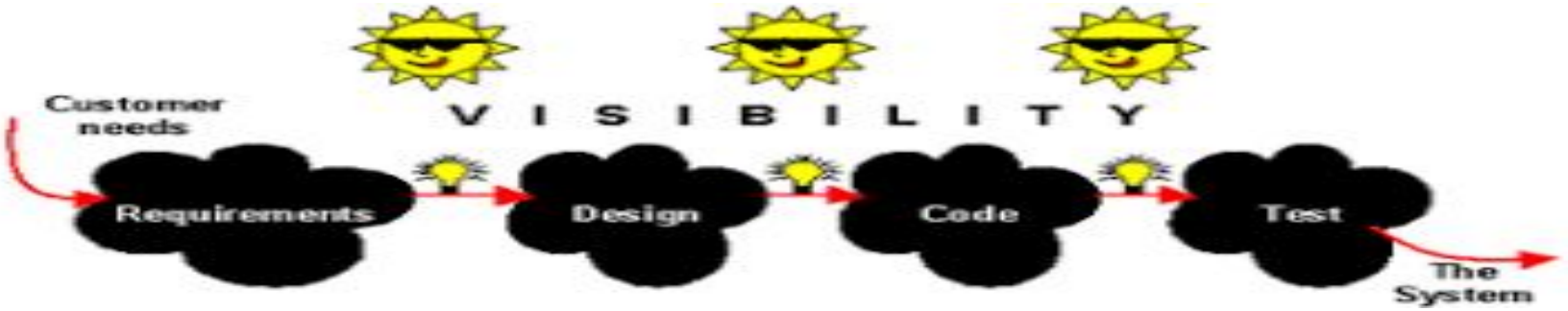
# Waterfall Model - Application

- **Requirements are very well documented, clear and fixed.**
- **Product definition is stable.**
- **Technology is understood and is not dynamic**.
- There are **no ambiguous(vague) requirements**.
- **Ample resources** with required expertise are available to support the product.
- **The project is short**.

# Waterfall Model - Advantages

- **Simple and easy to understand and use**
- **Easy to manage due to the rigidity of the model**. Each phase has specific deliverables and a review process.
- **Phases are processed and completed one at a time**.
- **Works well for smaller projects.**
- **Clearly defined stages**.
- **Well understood milestones**.[goals/auditing]
- **Easy to arrange tasks**.
- **Process and results are well documented**.

- A **milestone** is a significant event in the course of a project that is used to give visibility of progress in terms of achievement of predefined Milestones goals Failure to meet a **milestone** indicates that a project is not proceeding to plan and usually triggers corrective action by management.[ it's a kind of auditing ]



# Waterfall Model – **Disadvantages**

- **High amounts of risk and uncertainty.**
- **Not a good model for complex and object-oriented projects.**
- **Poor model for long and ongoing projects.**
- **Not suitable for the projects where requirements are at a moderate to high risk of changing.**
- It is **difficult to measure progress** within stages.
- **Cannot accommodate changing requirements.**
- **Adjusting scope during the life cycle can end a project.**
- **Integration is done as a "big-bang".**
- Real projects are rarely sequential

# Disadvantages of Waterfall Model

1. Time consuming
2. Never backward (Traditional)
3. Little room for iteration
4. Difficulty responding to change
5. Not suitable for the projects where requirements are at a moderate to high risk of changing
6. Product available at the end of the phase.

# **Big Bang** model -SDLC

# **Big Bang** model -SDLC

- The **Big Bang** model is an SDLC model where we do not follow any specific process.
  The **development** just starts with the required money and efforts as the input, and the output is the **software** developed which may or may not be as per customer requirement.
- Big bang model is focusing on all types of resources in software development and coding, with no or very little planning. The requirements are understood and implemented when they come.

# Big Bang Model – Design and Application

The Big Bang Model comprises of focusing all the possible resources in the software development and coding, with very little or no planning. The requirements are understood and implemented as they come. Any changes required may or may not need to revamp the complete software.

This model is ideal for small projects with one or two developers working together and is also useful for academic or practice projects. It is an ideal model for the product where requirements are not well understood and the final release date is not given.

The advantages of the Big Bang Model are as follows –

- This is a very simple model
- Little or no planning required
- Easy to manage
- Very few resources required
- Gives flexibility to developers
- It is a good learning aid for new comers or students.

The disadvantages of the Big Bang Model are as follows –

- Very High risk and uncertainty.
- Not a good model for complex and object-oriented projects.
- Poor model for long and ongoing projects.
- Can turn out to be very expensive if requirements are misunderstood.

- Waterfall Model,
- Prototype Model,
- Spiral Model,
- Evolutionary Development Models-
- Iterative model and incremental model

# Prototype Model

# Software Prototype Model

- What is Software Prototyping?

**Prototype is a working model of software with some limited functionality.**

**Prototyping is used to allow the users evaluate developer proposals and try them out before implementation.**

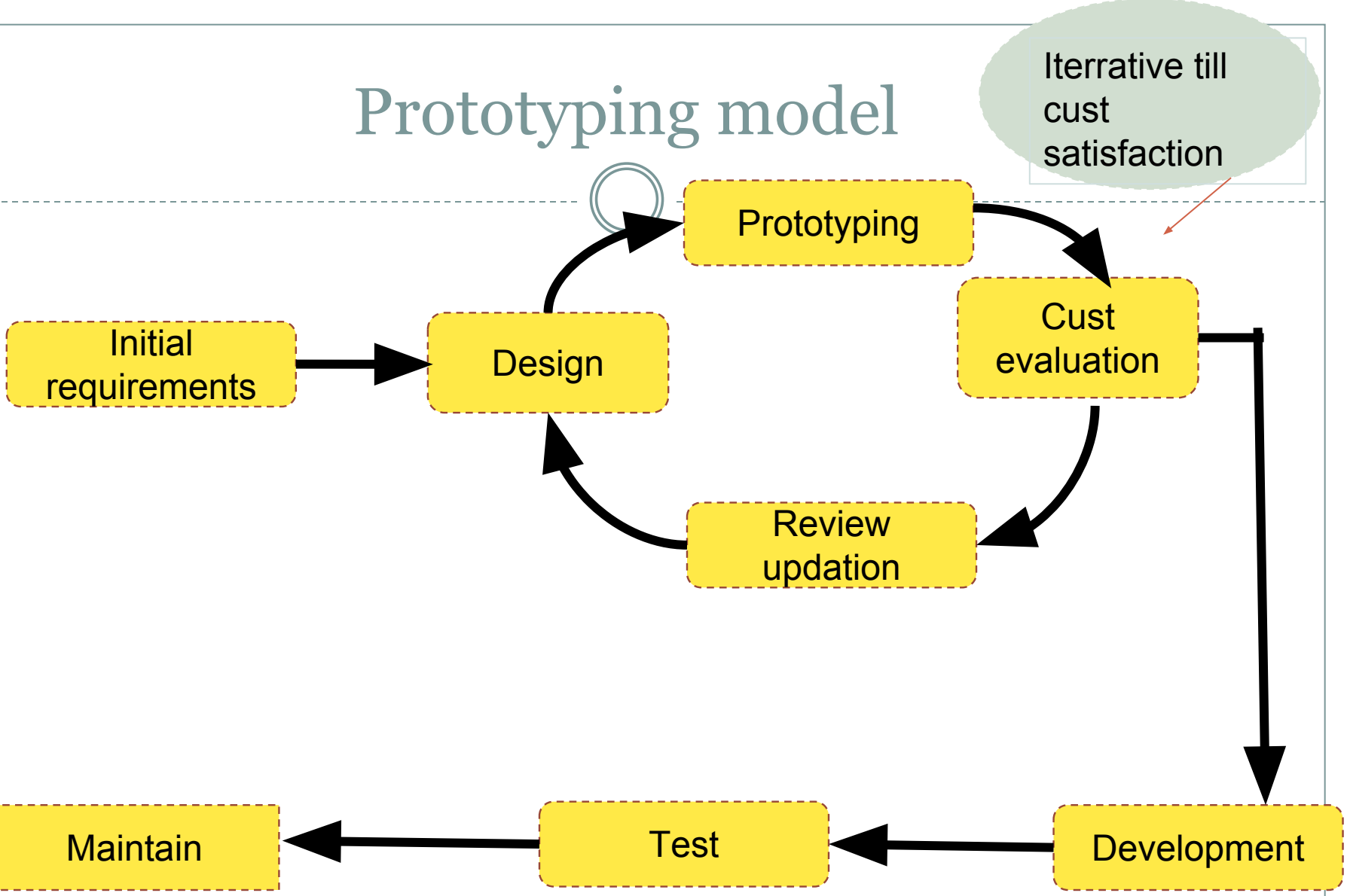
# What is Prototype Model?

- The Prototype model is one of the software development life cycle models in which a prototype is built with minimal requirements. This prototype is then tested and modified based on the feedback received from the client until a final prototype with desired functionalities gets created. This final prototype also acts as a base for the final product.

- The prototyping model starts with the requirements gathering. The developer and the user meet and define the purpose of the software, identify the needs, etc.
- A '**quick design**' is then created. This design focuses on those aspects of the software that will be visible to the user. It then leads to the development of a prototype. The customer then checks the prototype, and any modifications or changes that are needed are made to the prototype.

- Looping takes place in this step, and better versions of the prototype are created. These are continuously shown to the user so that any new changes can be updated in the prototype. This process continue until the customer is satisfied with the system. Once a user is satisfied, the prototype is converted to the actual system with all considerations for quality and security.

# Prototyping model



# Software Prototype Model

- **Following is a stepwise approach:**

1. Basic Requirement Identification
2. Developing the initial Prototype [iterative till cust satisfaction ]
    1. DESIGN
    2. PROTOTYPING
    3. CUST EVALUATION
    4. REVIEW UPADATION
3. Development
4. Testing
5. Maintenance

# Software Prototyping - Types

1. **Throwaway/Rapid/ close ended Prototyping**
2. **Evolutionary Prototyping**
3. **Incremental Prototyping** - to building multiple functional prototypes of the various sub-systems
4. **Extreme Prototyping**

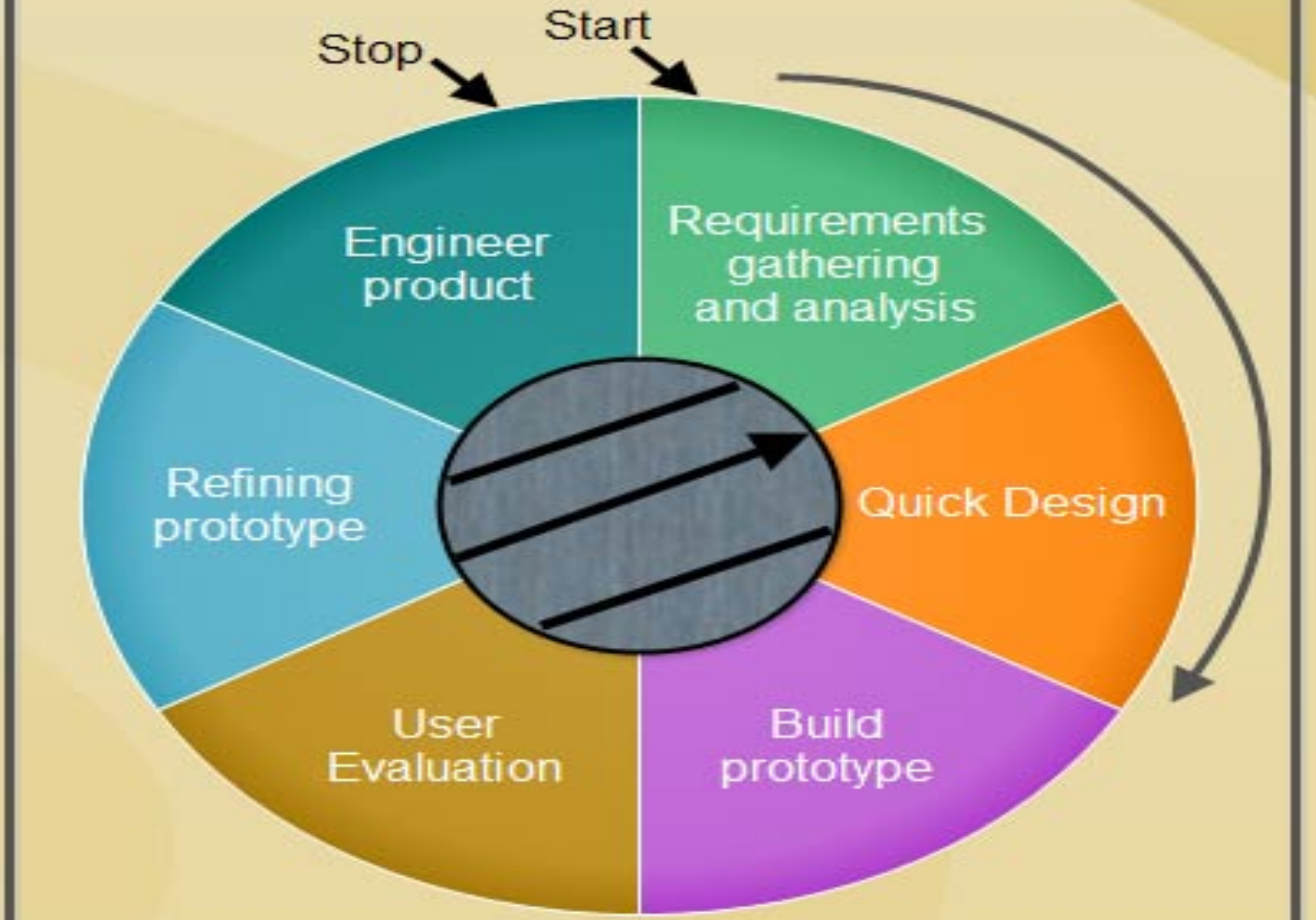
# Types of Prototype Model

- **Rapid Throwaway Prototyping** – In this method, the prototype is developed rapidly based on the initial requirements and given to the client for review. Once the client provides feedback, final requirements are updated and work on the final product begins. As the name suggests, the developed prototype is discarded, and it will not be part of the final product. It is also known as close-ended prototyping.

  **Evolutionary Prototyping –** In this method, a prototype is made, and the client feedback is received. Based on the feedback, the prototype is refined until the client considers it the final product. It follows an incremental development approach and saves time compared to the rapid throwaway prototyping method as in evolutionary prototyping old prototype is reworked rather than developing a new prototype from scratch. It is also known as breadboard prototyping.

- **Incremental Prototyping –** In this type of prototype model, final product requirements are break into smaller parts and each part is developed as a separate prototype. In the end, all the parts (prototypes) are merged which becomes a final product.

- **Extreme Prototyping** – This type of prototyping model is mainly used for web applications. It is divided into three phases–
  - First, a basic prototype with static pages is created, it consists of HTML pages.
  - Next, using a services layer, data processing is simulated.
  - In the last phase, services are implemented.


Stop
Start
Engineer product
Requirements gathering and analysis
Refining prototype
Quick Design
User Evaluation
Build prototype

# The advantages of the Prototyping Model

1. **Increased user involvement**
2. **the users get a better understanding** of the system being developed.[see steady progress ]
3. **Reduces time and cost** as the defects can be detected much earlier.
4. **Quicker user feedback is available** leading to better solutions.
5. **Missing functionality can be identified easily.**
6. **Confusing or difficult functions can be identified.**
7. **This is useful when requirements are changing rapidly**

# The Disadvantages of the Prototyping Model

1. **It is imposibe to know how long it will take**
2. **There is no way to know number of iterative will be required**
3. **Risk of insufficient requirement analysis**
4. **Users may get confused in the prototypes and actual systems.**
5. **Complex :the system may expand beyond original plans.**
6. **Reusing existing prototype may not be feasible always(developer)**
7. **The effort invested in building prototypes may be too much**

- Disadvantages

- It is a time-consuming process or method as multiple prototypes might be needed until the client reaches the final requirements. The Client may not have an explicit idea about what they want.
- This method involves too much client interaction and involvement, which can be done only with a committed client.
- In the beginning, it is a bit difficult to predict the exact amount of time needed to reach the final product.
- While coding, developers do not have a broad perspective of what is coming, because of which they might use an underlying architecture that is not suitable for a final product.
- To produce the quick prototype, developers might make weak decisions during the development process (especially implementation decisions), and compromise on quality which might eventually affect the product.

# Conclusion

- The prototype model is a trial and error method which has its advantages and disadvantages. It is particularly useful when the client does not have clarity on what all features, they need in the product.

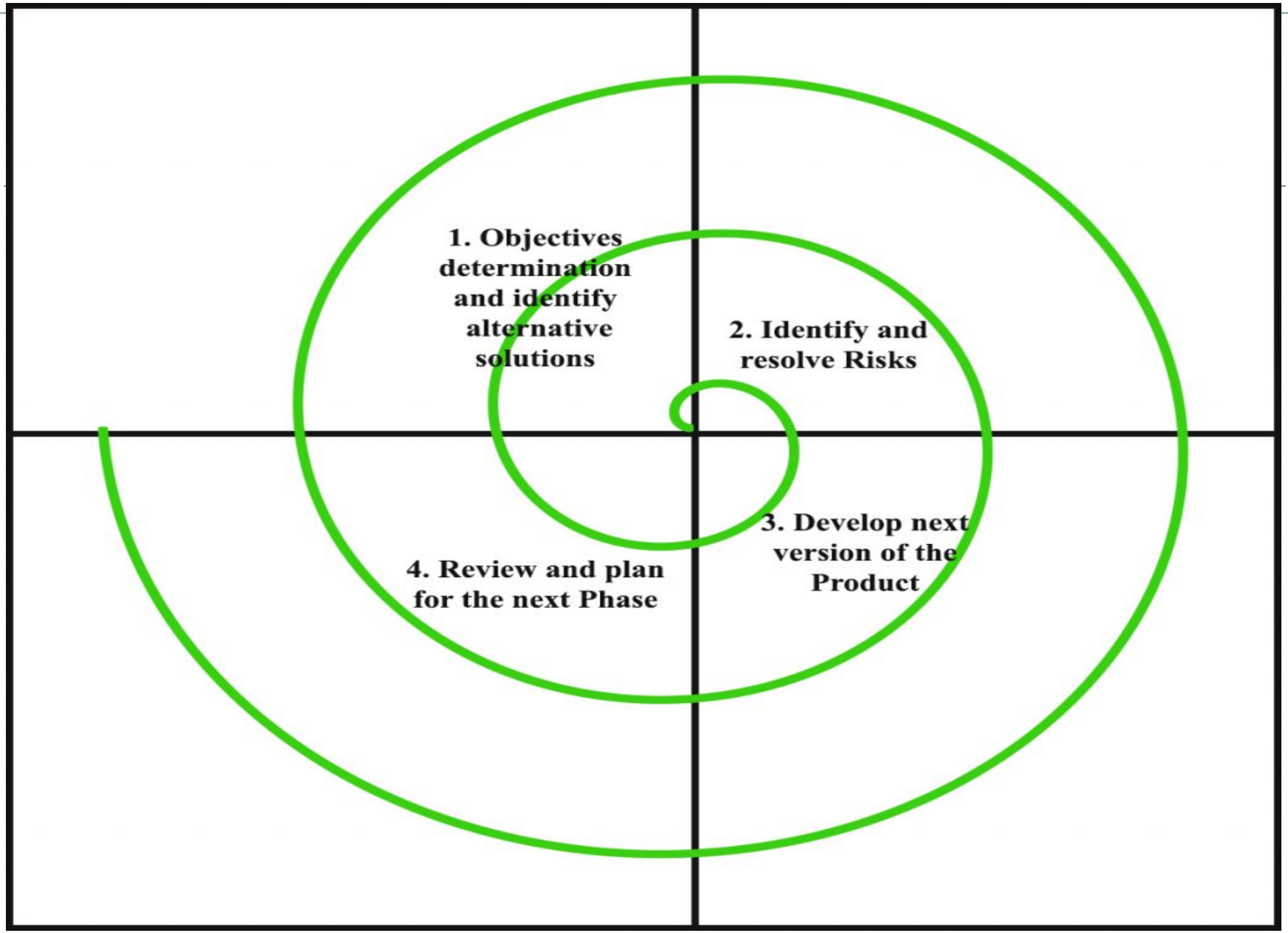
- Waterfall Model,
- Prototype Model,
- Spiral Model,
- Evolutionary Development Models-
- Iterative model and incremental model

# Spiral Model

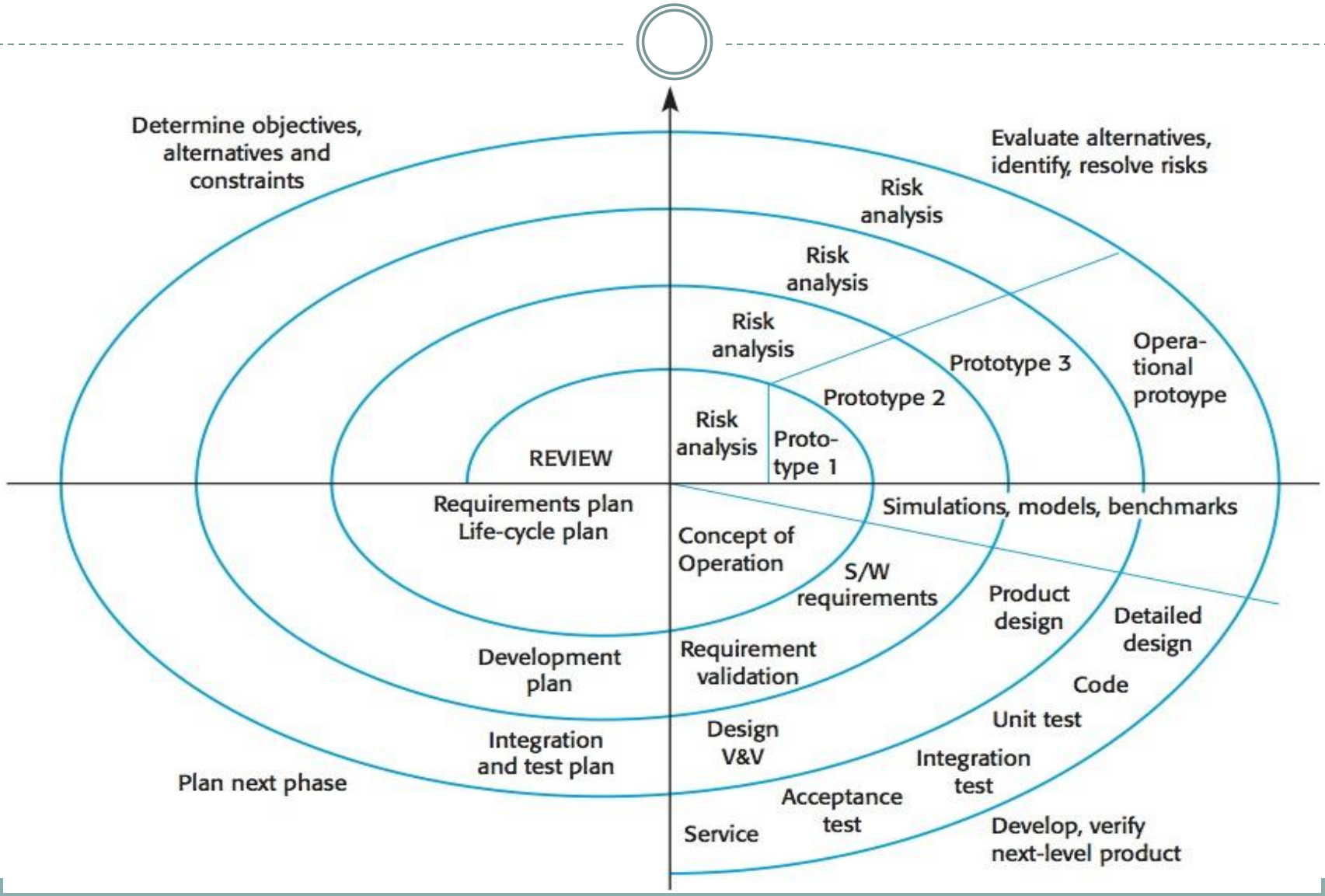
The spiral model, initially proposed by Boehm, is an evolutionary software process model that couples the iterative feature of prototyping with the controlled and systematic aspects of the linear sequential model. It implements the potential for rapid development of new versions of the software. Using the spiral model, the software is developed in a series of incremental releases. During the early iterations, the additional release may be a paper model or prototype. During later iterations, more and more complete versions of the engineered system are produced.

- he spiral model is a **risk-driven process model**. This SDLC model helps the group to adopt elements of one or more process models like a waterfall, incremental, waterfall, etc. The spiral technique is a combination of rapid prototyping and concurrency in design and development activities.
- Each cycle in the spiral begins with the identification of objectives for that cycle, the different alternatives that are possible for achieving the goals, and the constraints that exist. This is the first quadrant of the cycle (upper-left quadrant).

- The next step in the cycle is to evaluate these different alternatives based on the objectives and constraints. The focus of evaluation in this step is based on the risk perception for the project.
- The next step is to develop strategies that solve uncertainties and risks. This step may involve activities such as benchmarking, simulation, and prototyping.


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

# Spiral Model



- The functions of these four quadrants are discussed below-

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.

- Spiral Model is called **meta-model** because **it uses both Waterfall and Prototype models.**

- **The whole development process repeatedly passes trough these stages**.

- **Each iteration is called Spiral.**

- Four main phases are:

✔ Identification
[Determine Objectives /resources/risks/statkeholders )

✔ Design
requirements are proportional to the design

✔ Build / construct
code and test

✔ Evaluation / risk analysis
risk identified and eliminated in every spiral/ iterrative

- It will be used for high risk projects
- Combination of iterative and waterfall
- Iterative for four phases only
- Once all spiral finished , means  the cust will be happy
- No risk , no fault , no error then the s/w will be deploy and maintenance

## When to use Spiral Model?

- When deliverance is required to be frequent.
- When the project is large
- When requirements are unclear and complex
- When changes may require at any time
- Large and high budget projects

# Spiral Model Applications

- When there is a **budget constraint** and **risk evaluation** is important.
- **For medium to high-risk projects**.
- **Needs Long-term project commitment**
- **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.**

# Spiral Model Advantages

- **Changing requirements can be accommodated in later stages.**
- **Allows extensive use of prototypes.**
- **Requirements can be captured more accurately.**
- **Users see the system early/project monitoring is easy**
- **Development can be divided into smaller parts and the risky parts can be developed earlier which helps in better risk management.**
- **Suitable for high risk project and reduces number of risk**

# Spiral Model Disadvantages

- **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**.
- Strict rules and protocols are followed

- Waterfall Model,
- Prototype Model,
- Spiral Model,
- Evolutionary Development Models-
- Iterative model and incremental model

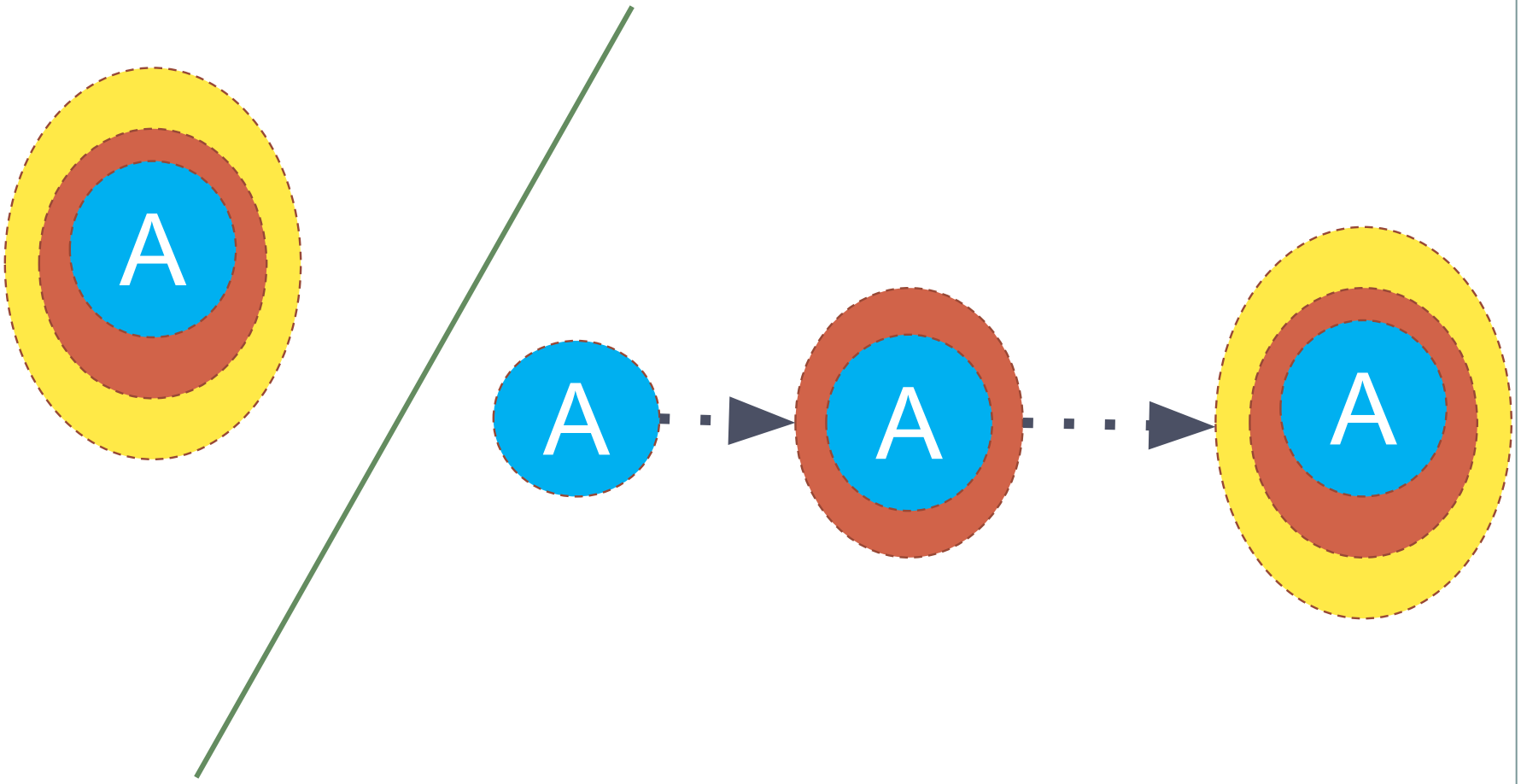
# Evolutionary development model

# Evolutionary model

- This model is known as successive model. Here system is broken into several model and successfully delivered
- As it is incremental model each page of this model is busy until the product is delivered[ eg module A is busy ]
- First of all the core model is developed and then it is added into the increasing levels with another model

  [ model A  in 1st phase and 2nd and3rd ]

# Evolutionary model



## Advantages

- User can easily understood the system and experiment with partially developed system
- Useful in identifying requirement
- It reduces the chance of error by testing core model multiple times

## Dis advantages

- If a problem is not easy to divide in module then this model cannot be implemented correctly

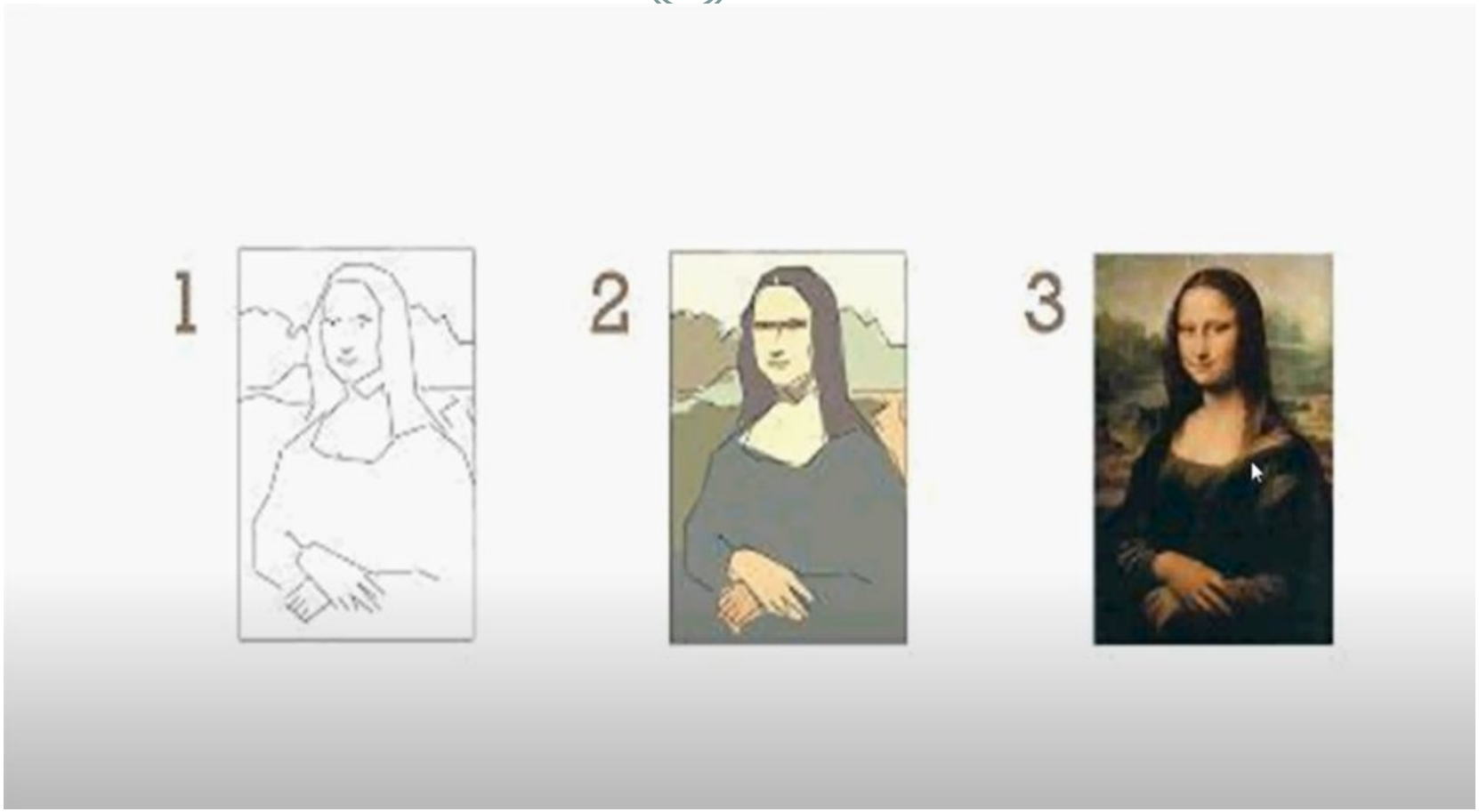
- Waterfall Model,
- Prototype Model,
- Spiral Model,
- Evolutionary Development Models-
- Iterative model and incremental model

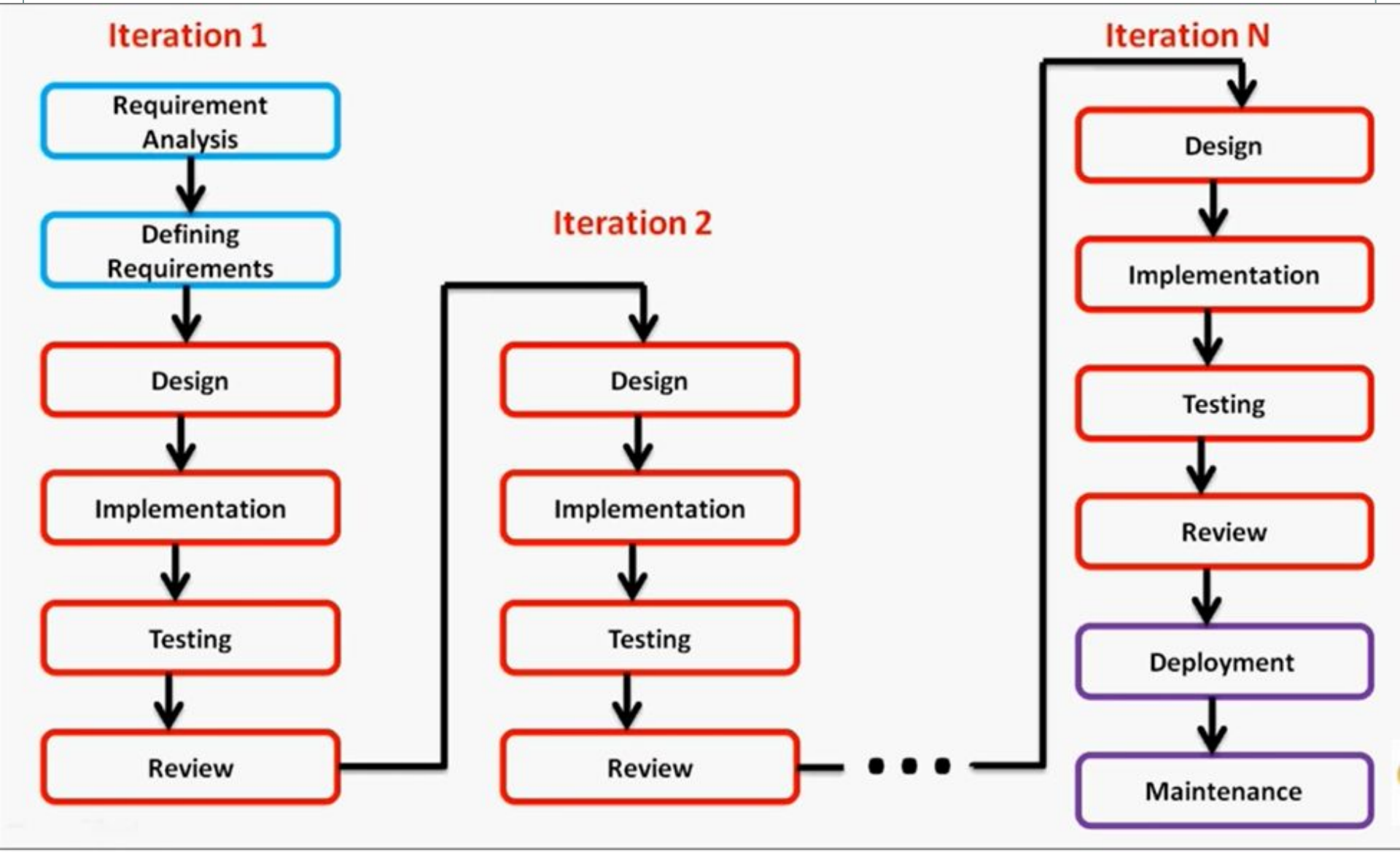
# Iterative model

# Iterative Model

In **iterative model** the organization start with some of the software specification and develop the first version of the software. After the first version if there is a need to change the software then a new version of the software is created with a new iteration.
It will repeat until deployment of the software.

# Iterations –sketches , few color and finish




Iteration 1
Requirement Analysis
Defining Requirements
Design
Implementation
Testing
Review
Iteration 2
Design
Implementation
Testing
Review
Iteration N
Design
Implementation
Testing
Review
Deployment
Maintenance

## When to use the Iterative Model

The iterative model is suitable for the following use cases:

- When the project is huge, it can be broken down into smaller pieces and developed by adhering to the iterative paradigm.
- When the requirements can be understood and defined clearly at the beginning of the project.
- When there is a need to incorporate customer feedback at every stage - The major requirements are laid down initially; however, as the development process progresses, some functionalities are altered, and additions are suggested.
- While working on the project, the development team is experimenting with and learning new technology.

# Advantages of the Iterative Model

- The most important advantage of this approach is that software is built in iterations, allowing developers and testers to identify design or functionality faults as quickly as possible, allowing them to take corrective actions within a limited budget. Here is a list of advantages of this SDLC model:

- A working product is produced much early in the lifecycle, unlike the waterfall model, where a working product is available only at the end of the lifecycle.

- We can detect errors and bugs at an early stage and prevent them from flowing downwards. We test the output of every iteration and do not let bugs from the previous iteration propagate to the next iteration.

- Changing the requirements does not incur much cost in this model, although it may not always be possible to accommodate new requirements due to system structure and design constraints.
- Customer feedback can be incorporated in every iteration and implemented quickly.
- In this model, less time is spent on documenting and more time on designing and developing.

## Disadvantages of the Iterative Model

- Although the iterative model is hugely beneficial, it has various faults and drawbacks. The biggest one is system structure or design issues may arise since not all requirements are collected at the advent of the entire life cycle. The disadvantages of the iterative model are listed below:

- Problems relating to the system architecture can arise because all the requirements are not gathered upfront. Design can be changed repeatedly because of defective requirements gathered in the first phase.

- Although the cost of change is lower, it is not well suited to frequently shifting requirements.
- It is not a good option for small projects as it may not be realistic or possible to break down a small project into smaller parts.
- It is more resource-intensive than the waterfall model. To avoid risk, this strategy necessitates the use of highly trained resources for the project's analysis.
- The entire procedure is challenging to manage.

## Examples

1. The iterative model is prevalent in the area of electronics. The evolution of mobile phones over time, how speakers have become smaller and more portable over time, or even how refrigerators from the same company have altered to accommodate new family needs. These are all iterative procedures. Every day, the industry is changing itself in response to client feedback.

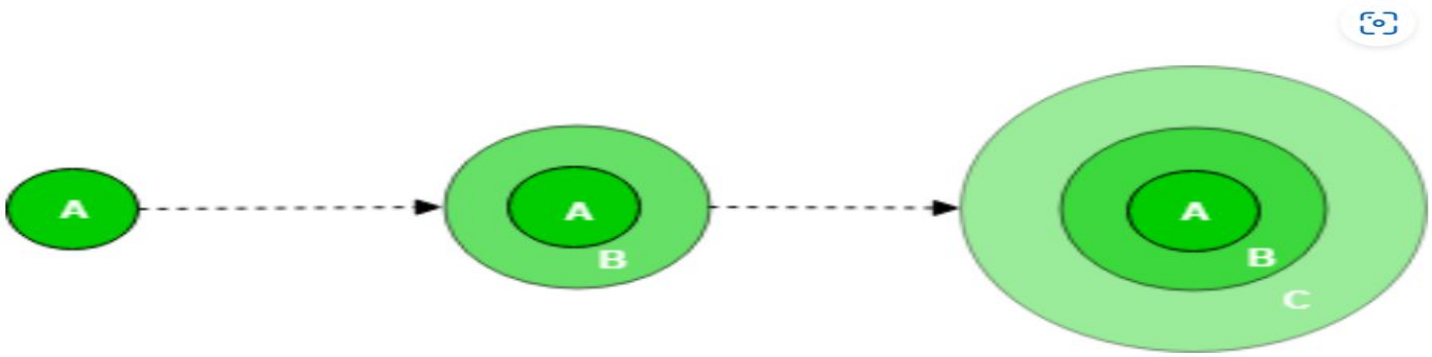
2. Digital marketing teams test different advertising methods to discover which one generates the most engagement. They comprehend the advertisement's requirements, build appealing designs based on thorough analysis, and implement them in their product marketing. These copies are then distributed to members for input, and modifications are made. This strengthens their marketing strategies.

# Incremental Model

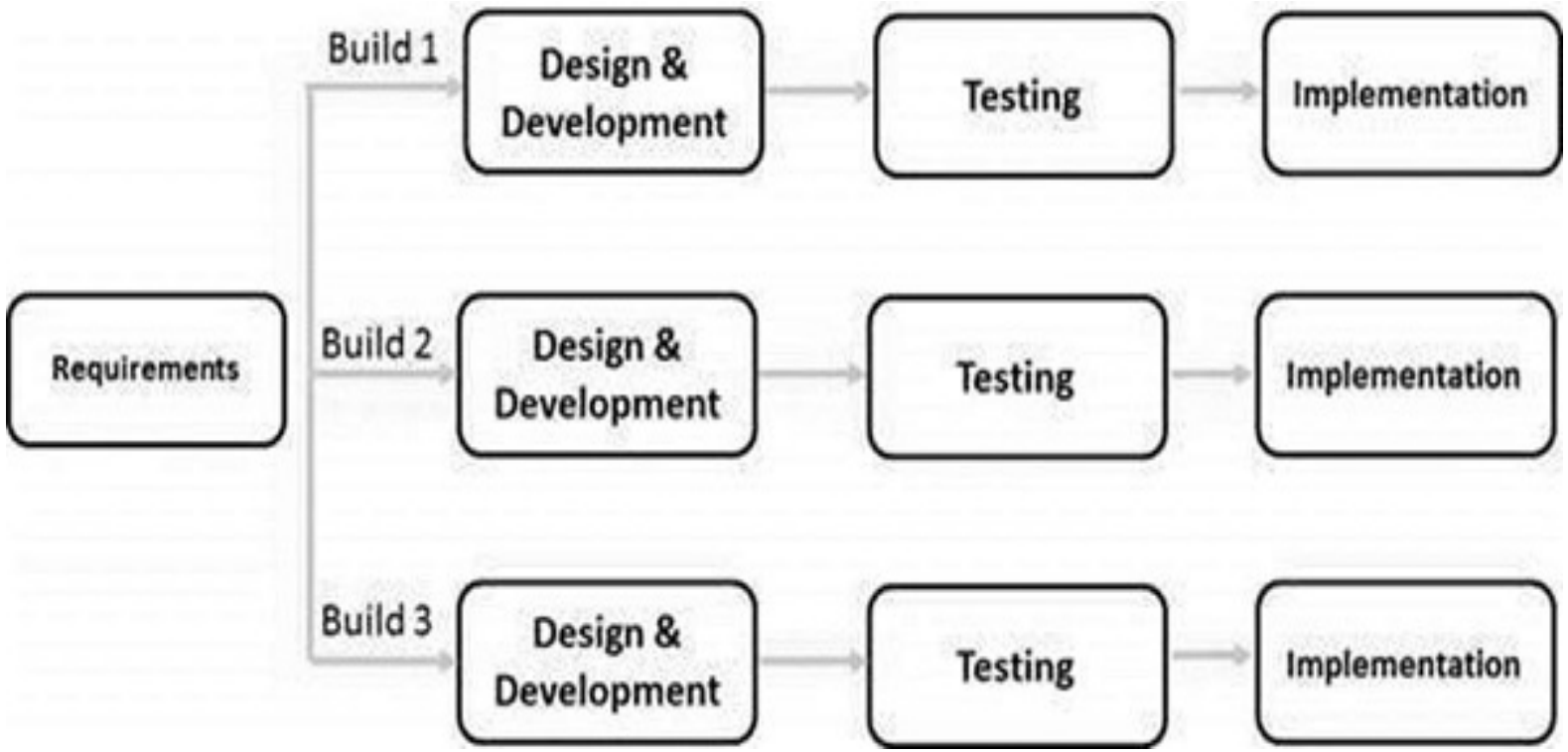
- Waterfall Model,
- Prototype Model,
- Spiral Model,
- Evolutionary Development Models-
- Iterative model and incremental model

- 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. Every subsequent release of the module adds function to the previous release. The process continues until the complete system achieved.

- The incremental process model is also known as the Successive version model.
- First, a simple working system implementing only a few basic features is built and then that is delivered to the customer. Then thereafter many successive iterations/ versions are implemented and delivered to the customer until the desired system is released.



- A, B, and C are modules of Software Products that are incrementally developed and delivered.


Requirements
Build 1
Design & Development
Testing
Implementation
Build 2
Design & Development
Testing
Implementation
Build 3
Design & Development
Testing
Implementation

- The basic idea behind this method is:

to develop a system through repeated cycles (iterative) and in smaller portions at a time (incremental).

**more than one iteration** of the software development cycle **may be in progress** at the same time."

# advantages

- Some working functionality can be developed quickly
- Results are obtained early and periodically.
- Parallel development can be planned.
- Progress can be measured.
- **Less costly to change the scope/requirements**.
- **Testing and debugging during smaller iteration** is easy.
- **Risks are identified and resolved during iteration**; and each iteration is an easily managed milestone.

# dis advantages

- **More resources may be required**.
- Although cost of change is lesser, but it is not very suitable for changing requirements.
- **More management attention is required.**
- **System architecture or design issues may arise**

# dis advantages

- **Defining increments may require definition of the complete system.**
- Not suitable for smaller projects.
- **End of project may not be known which is a risk**.
- Highly skilled resources are required for risk analysis.

# DIFFERENCE BW IM AND IM

- The iterative method works by sectioning the process into individual, fully functional development cycles or "sprints.” This allows the project team to produce a complete product before trying out another more improved iteration of it. In contrast, the incremental method operates by building the entire product in small incomplete parts or increments, with a fully functional version only accessible at the end of the development process when all the unfinished sections come together.

Any questions

| Classical waterfall | Iterative waterfall | Prototype Model | Incremental Model | Evolutionary Model | RAD Model | Spiral Model | Agile Model |
|---|---|---|---|---|---|---|---|
| Basic, Rigid, Inflexible, Not for Real Project | Basic, Problem is well understood | User Requirement Not clear, Costly, No Early look on Requirements → High User Involvement → Reusability | Module by Module Delivery, Easy to test and debug | Large Projects | Time and Cost Constraint, User at all levels → Reusability | Risk, Not for small Projects, → No Early look on Requirements, → Less Experience Can work | Flexible, Advanced, Parallel, Process divided into sprints |

# Q B

- What is SDLC
- Explain briefly sdlc in software engineering
- What is the difference between iterative and incremental model
- **Exaplain linear sequential model** or **classic Life cycle Model**.
- Explain spiral model