# Unit 1
# Software Engineering

**Prepared By**

**Abhishek Kesharwani**

**Assistant Professor ,United College of Engineering and Research**

## Index

## Software Development Life Cycle (SDLC)

- A software life cycle model (also termed process model) is a pictorial and diagrammatic representation of the software life cycle.
- A life cycle model represents all the methods required to make a software product transit through its life cycle stages.
- It also captures the structure in which these methods are to be undertaken.
- In other words, a life cycle model maps the various activities performed on a software product from its inception to retirement. Different life cycle models may plan the necessary development activities to phases in different ways.
- Thus, no element which life cycle model is followed, the essential activities are contained in all life cycle models though the action may be carried out in distinct orders in different life cycle models.
- During any life cycle stage, more than one activity may also be carried out.

## Need of SDLC

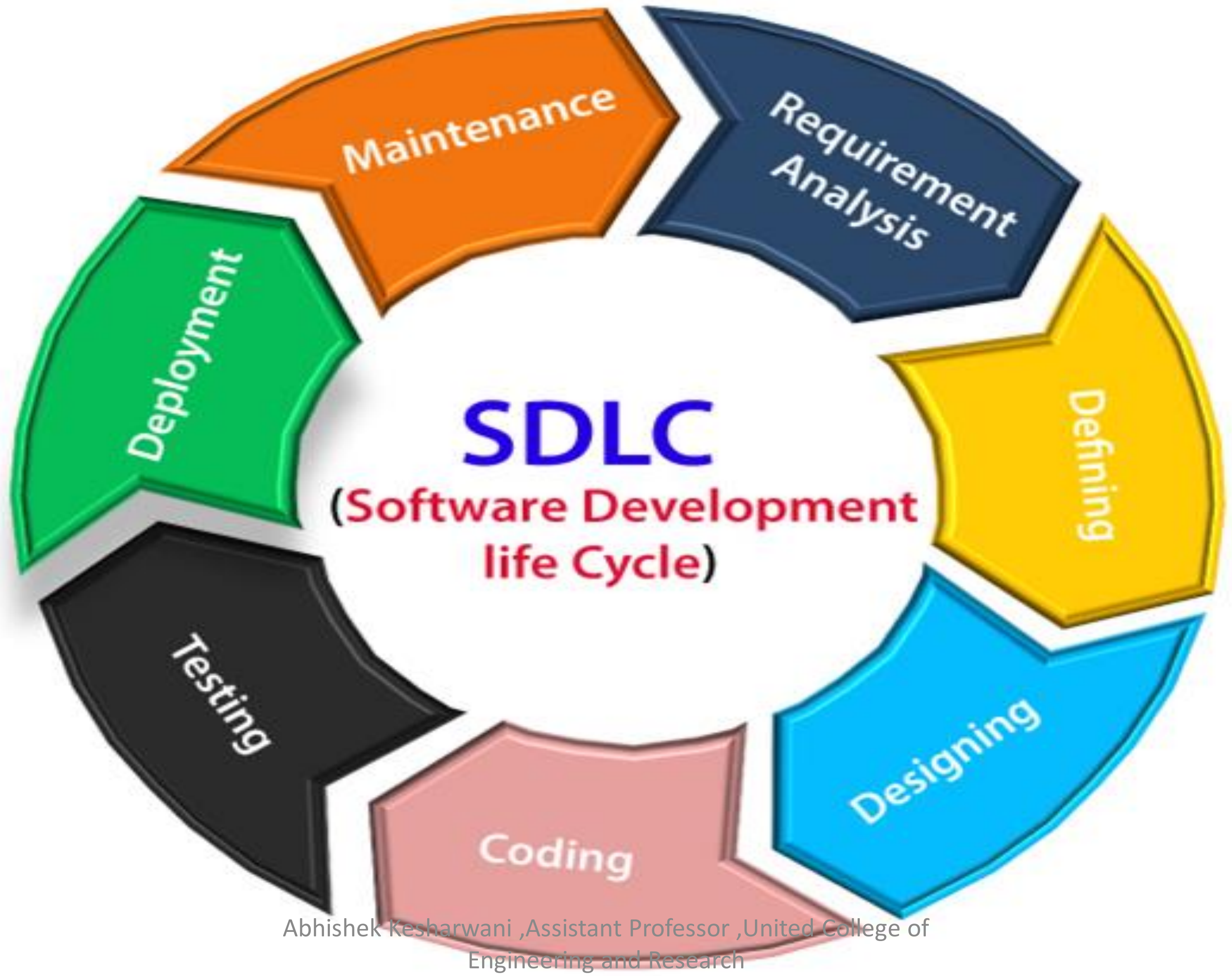
- The development team must determine a suitable life cycle model for a particular plan and then observe to it.
- Without using an exact life cycle model, the development of a software product would not be in a systematic and disciplined manner.
- When a team is developing a software product, there must be a clear understanding among team representative about when and what to do. Otherwise, it would point to chaos and project failure.

This problem can be defined by using an example.

- Suppose a software development issue is divided into various parts and the parts are assigned to the team members.
- From then on, suppose the team representative is allowed the freedom to develop the roles assigned to them in whatever way they like. It is possible that one representative might start writing the code for his part, another might choose to prepare the test documents first, and some other engineer might begin with the design phase of the roles assigned to him.
- This would be one of the perfect methods for project failure.

- A software life cycle model describes entry and exit criteria for each phase.
- A phase can begin only if its stage-entry criteria have been fulfilled. So without a software life cycle model, the entry and exit criteria for a stage cannot be recognized.
- Without software life cycle models, it becomes tough for software project managers to monitor the progress of the project.

## SDLC Cycle represents the process of developing software. SDLC framework includes the following steps:



## The stages of SDLC are as follows:

### Stage1: Planning and requirement analysis

- Requirement Analysis is the most important and necessary stage in SDLC.
- The senior members of the team perform it with inputs from all the stakeholders and domain experts or SMEs in the industry.
- Planning for the quality assurance requirements and identifications of the risks associated with the projects is also done at this stage.
- Business analyst and Project organizer set up a meeting with the client to gather all the data like what the customer wants to build, who will be the end user, what is the objective of the product. Before creating a product, a core understanding or knowledge of the product is very necessary.

**For Example**, A client wants to have an application which concerns money transactions. In this method, the requirement has to be precise like what kind of operations will be done, how it will be done, in which currency it will be done, etc.

- Once the required function is done, an analysis is complete with auditing the feasibility of the growth of a product. In case of any ambiguity, a signal is set up for further discussion.
- Once the requirement is understood, the SRS (Software Requirement Specification) document is created. The developers should thoroughly follow this document and also should be reviewed by the customer for future reference.

## Stage2: Defining Requirements

- Once the requirement analysis is done, the next stage is to certainly represent and document the software requirements and get them accepted from the project stakeholders.
- This is accomplished through "SRS"- Software Requirement Specification document which contains all the product requirements to be constructed and developed during the project life cycle.

## Stage3: Designing the Software

- The next phase is about to bring down all the knowledge of requirements, analysis, and design of the software project. This phase is the product of the last two, like inputs from the customer and requirement gathering.

## Stage4: Developing the project

- In this phase of SDLC, the actual development begins, and the programming is built. The implementation of design begins concerning writing code.
- Developers have to follow the coding guidelines described by their management and programming tools like compilers, interpreters, debuggers, etc. are used to develop and implement the code.

## Stage5: Testing

- After the code is generated, it is tested against the requirements to make sure that the products are solving the needs addressed and gathered during the requirements stage.
- During this stage, unit testing, integration testing, system testing, acceptance testing are done.

## Stage6: Deployment

- Once the software is certified, and no bugs or errors are stated, then it is deployed.
- Then based on the assessment, the software may be released as it is or with suggested enhancement in the object segment.
- After the software is deployed, then its maintenance begins.

## Stage7: Maintenance

- Once when the client starts using the developed systems, then the real issues come up and requirements to be solved from time to time.

## Why SDLC?

Various reasons for using a life-cycle model include:-

– Helps to **understand the entire process**

– Enforces a **structured approach** to development

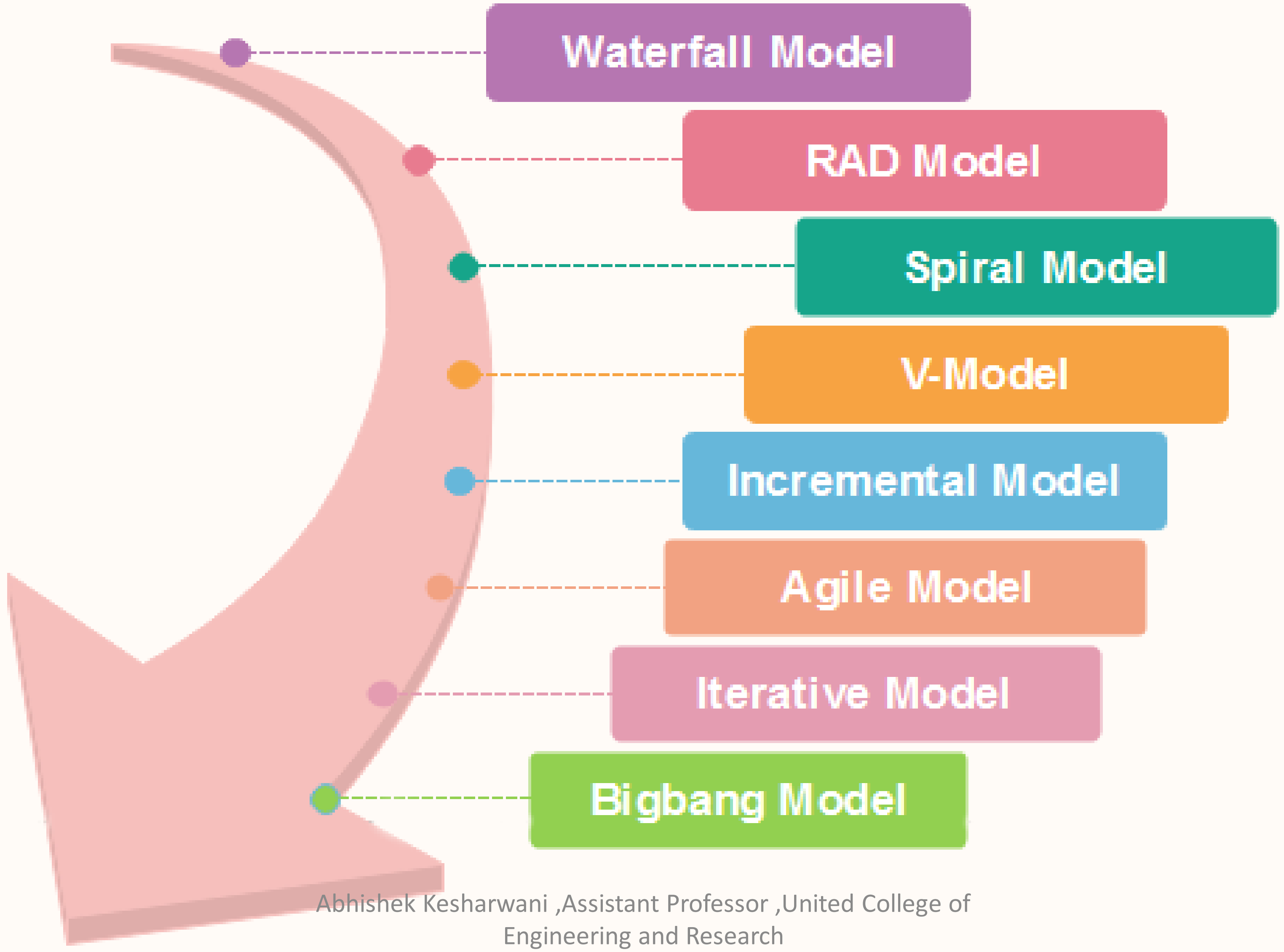
– Enables **planning of resources** in advance

– Aids management to **track progress of the system**

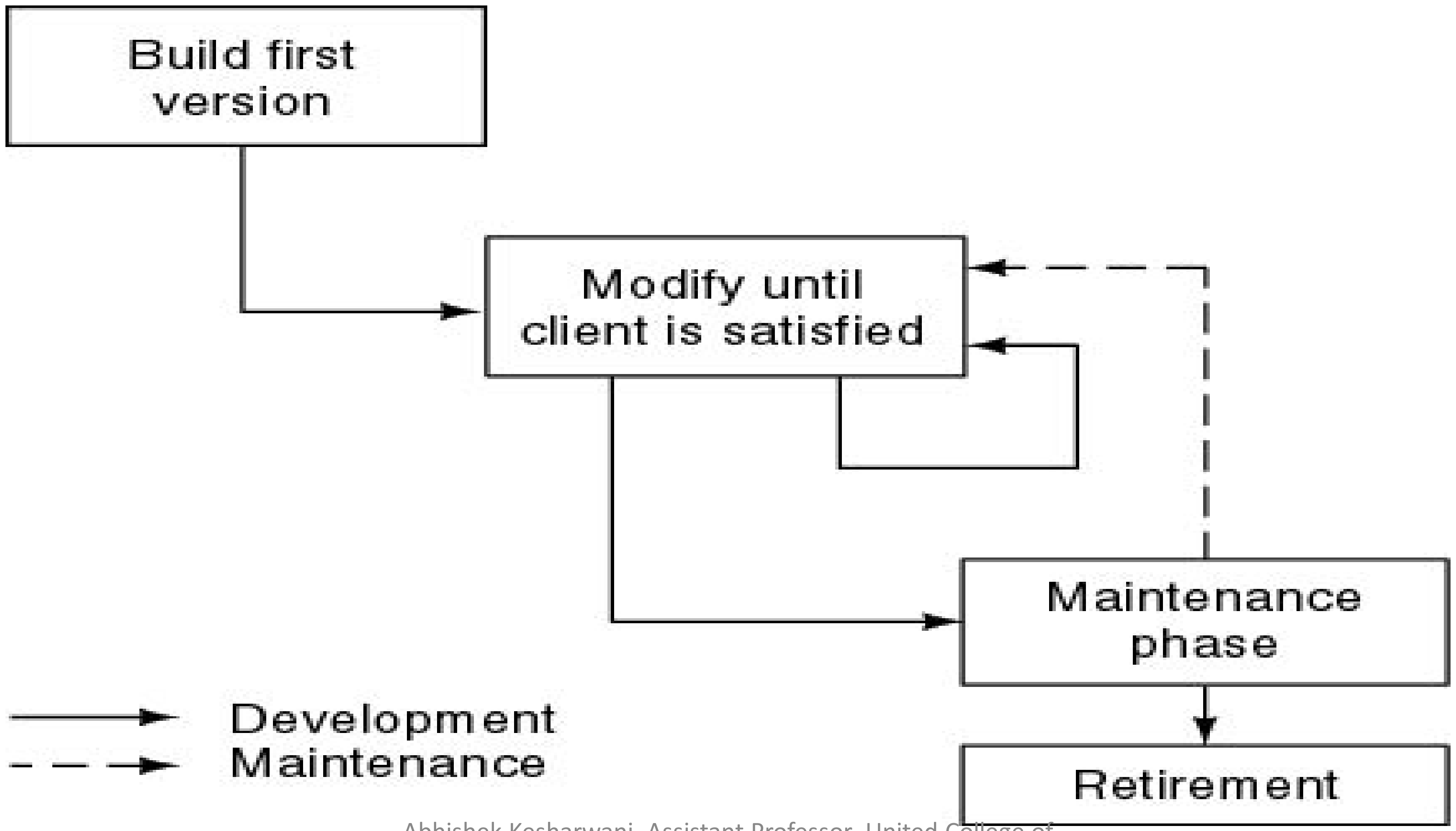
## SDLC Models

- 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

## Build and Fix Model



## Build and Fix Model

- This is the worst model for developing a project
- This model includes the following two phases.

  **Build:** In this phase, the software code is developed and passed on to the next phase.

  **Fix:** In this phase, the code developed in the build phase is made error free. Also, in addition to the corrections to the code, the code is modified according to the user's requirements.
- This model is completely unsatisfactory and should not be adopted.

- The software is developed without any specification or design.
- An initial product is built, which is then repeatedly modified until it (software) satisfies the user. That is, the software is developed and delivered to the user.
- The user checks whether the desired functions are present. If not, then the software is changed according to the needs by adding, modifying or deleting functions.
- This process goes on until the user feels that the software can be used productively. However, the lack of design requirements and repeated modifications result in loss of acceptability of software.
- Thus, software engineers are strongly discouraged from using this development approach.

## Advantages of Build and Fix Model

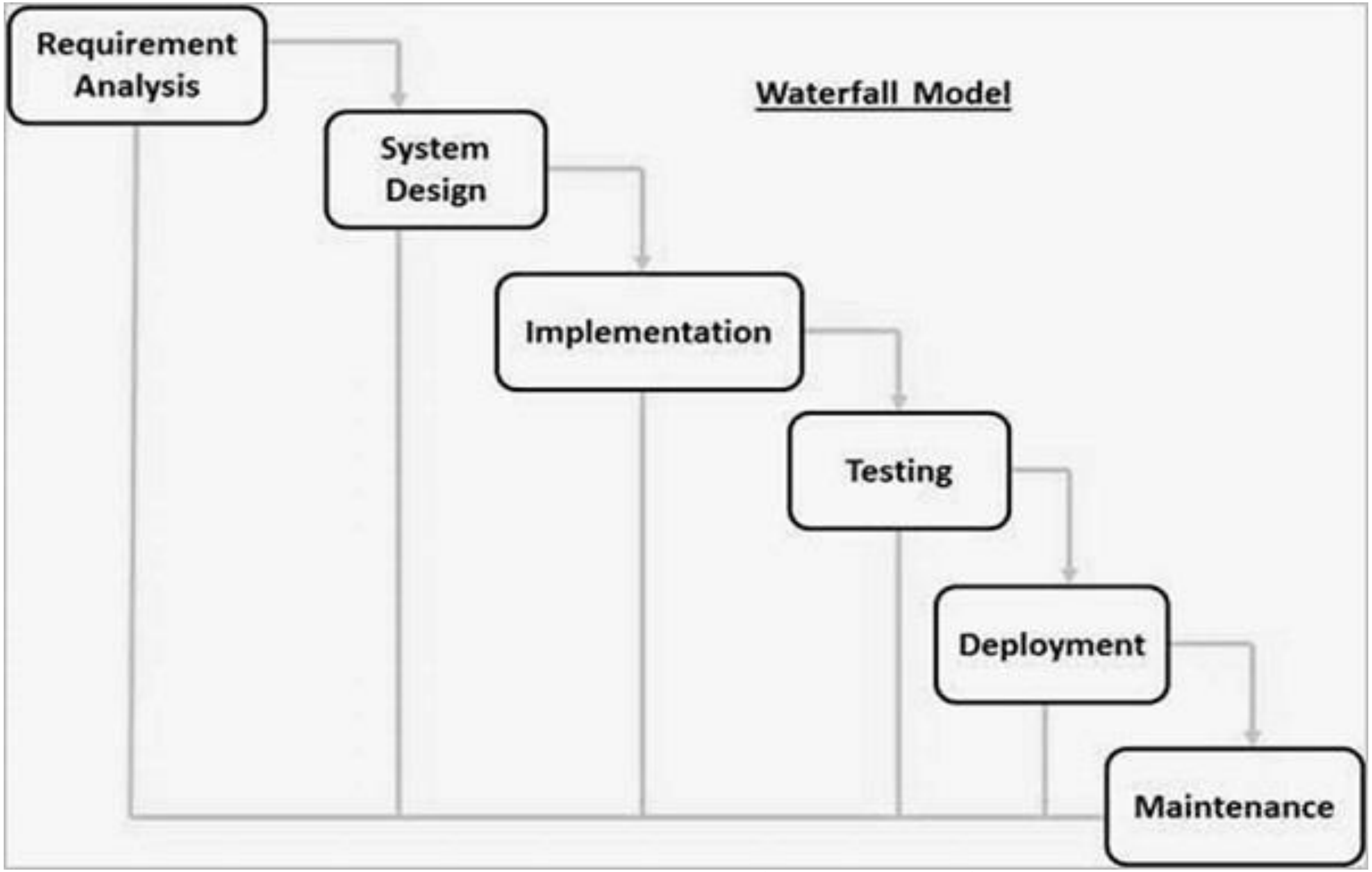
- **Requires less experience** to execute or manage other than the ability to program.
- **Suitable for smaller software.**
- **Requires less project planning.**

## Disadvantages of Build and Fix Model

- No real means is available of assessing the progress, quality, and risks.
- **Cost of using this process model is high** as it requires rework until user's requirements are accomplished.
- **Informal design** of the software as it involves unplanned procedure.
- **Maintenance of these models is problematic.**

## Waterfall model

- Winston Royce introduced the Waterfall Model in 1970.This model has five phases: Requirements analysis and specification, design, implementation, and unit testing, integration and system testing, and operation and maintenance.
- The steps always follow in this order and do not overlap.
- The developer must complete every phase before the next phase begins.
- This model is named "**Waterfall Model**", because its diagrammatic representation resembles a cascade of waterfalls.


Requirement Analysis
Waterfall Model
System Design
Implementation
Testing
Deployment
Maintenance

## 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.

All these phases are cascaded to each other in which progress is seen as flowing steadily downwards (like a waterfall) through the phases. The next phase is started only after the defined set of goals are achieved for previous phase and it is signed off, so the name "Waterfall Model". In this model, phases do not overlap

## 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 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 where requirements are very well understood.
- Clearly defined stages.
- Well understood milestones.
- Easy to arrange tasks.
- Process and results are well documented.

## Waterfall Model - Disadvantages

- No working software is produced until late during the life cycle.
- 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. So, risk and uncertainty is high with this process model.
- 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. at the very end, which doesn't allow identifying any technological or business bottleneck or challenges early.

## Iterative Model

- In this Model, you can start with some of the software specifications 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.
- Every release of the Iterative Model finishes in an exact and fixed period that is called iteration.
- The Iterative Model allows the accessing earlier phases, in which the variations made respectively.
- The final output of the project renewed at the end of the Software Development Life Cycle (SDLC) process.

## Iterative Model

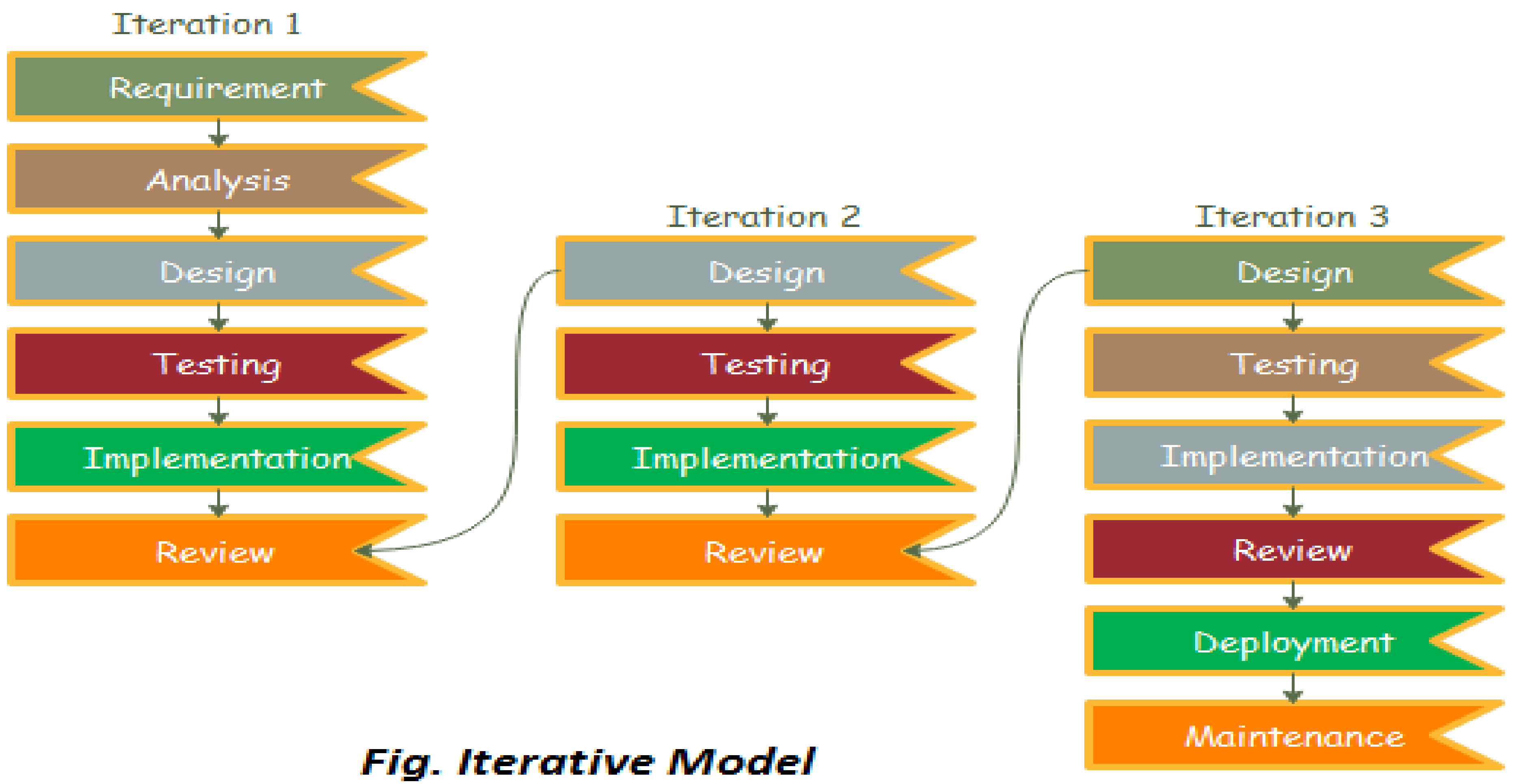


***Fig. Iterative Model***

## The various phases of Iterative model are as follows:

**1. Requirement gathering & analysis:** In this phase, requirements are gathered from customers and check by an analyst whether requirements will fulfil or not. Analyst checks that need will achieve within budget or not. After all of this, the software team skips to the next phase.

**2. Design:** In the design phase, team design the software by the different diagrams like Data Flow diagram, activity diagram, class diagram, state transition diagram, etc.

**3. Implementation:** In the implementation, requirements are written in the coding language and transformed into computer programmes which are called Software.

**4. Testing:** After completing the coding phase, software testing starts using different test methods. There are many test methods, but the most common are white box, black box, and grey box test methods.

**5. Deployment:** After completing all the phases, software is deployed to its work environment.

**6. Review:** In this phase, after the product deployment, review phase is performed to check the behaviour and validity of the developed product. And if there are any error found then the process starts again from the requirement gathering.

**7. Maintenance:** In the maintenance phase, after deployment of the software in the working environment there may be some bugs, some errors or new updates are required. Maintenance involves debugging and new addition options.

## When to use the Iterative Model?

- When requirements are defined clearly and easy to understand.
- When the software application is large.
- When there is a requirement of changes in future.

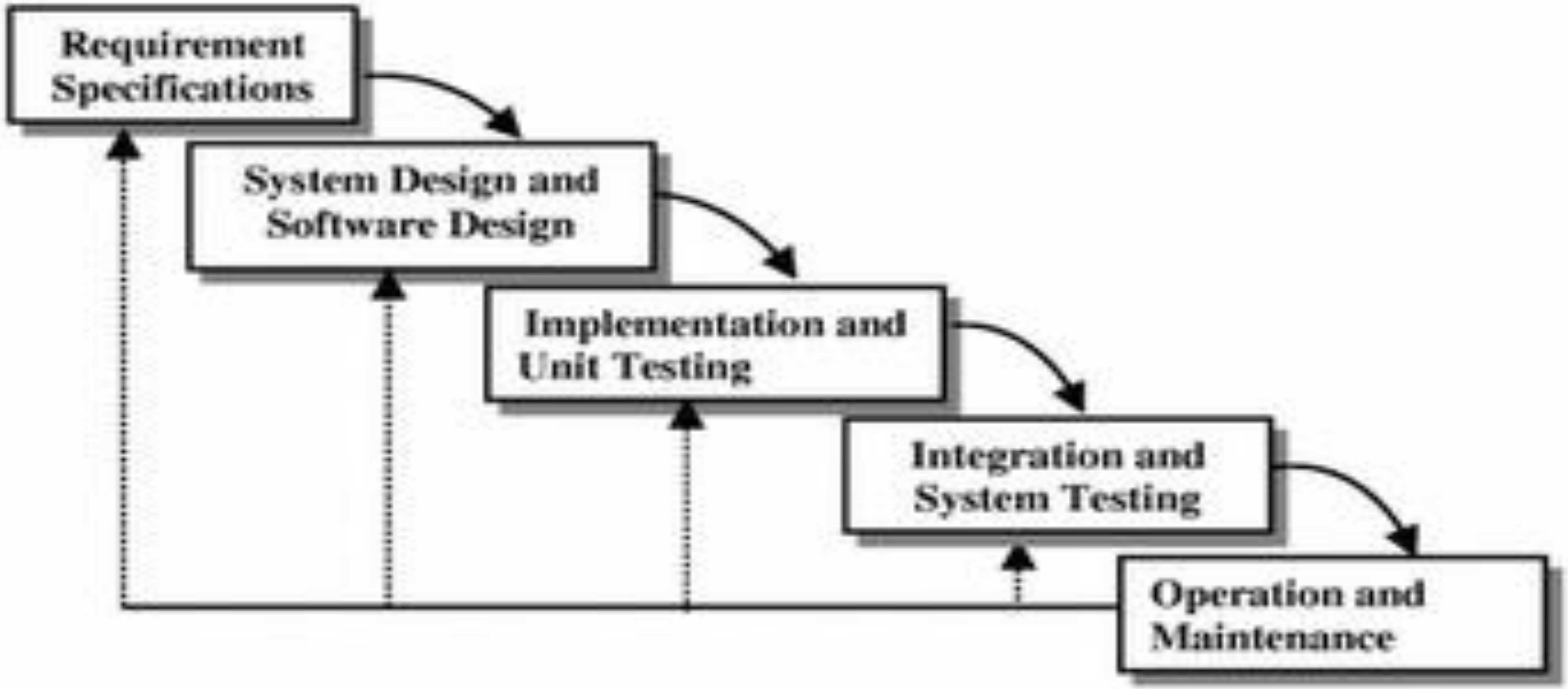
## Advantage(Pros) of Iterative Model

- Testing and debugging during smaller iteration is easy.
- A Parallel development can plan.
- It is easily acceptable to ever-changing needs of the project.
- Risks are identified and resolved during iteration.
- Limited time spent on documentation and extra time on designing.

## Disadvantage(Cons) of Iterative Model:

- It is not suitable for smaller projects.
- More Resources may be required.
- Design can be changed again and again because of imperfect requirements.
- Requirement changes can cause over budget.
- Project completion date not confirmed because of changing requirements.

# Iterative Enhancement Model


Requirement Specifications
System Design and Software Design
Implementation and Unit Testing
Integration and System Testing
Operation and Maintenance

- The incremental model (also known as **iterative enhancement model)** comprises the features of waterfall model in an iterative manner.
- The waterfall model performs each phase for developing complete software whereas the incremental model has phases similar to the linear sequential model arid has an iterative nature of prototyping.
- During the implementation phase, the project is divided into small subsets known as **increments** that are implemented individually. This model comprises several phases where each phase produces an increment.

- These increments are identified in the beginning of the development process and the entire process from requirements gathering to delivery of the product is carried out for each increment.
- The basic idea of this model is to start the process with requirements and iteratively enhance the requirements until the final software is implemented.
- In addition, as in prototyping, the increment provides feedback from the user specifying the requirements of the software.

- This approach is useful as it simplifies the software development process as implementation of smaller increments is easier than implementing the entire system.
- Each stage of incremental model adds some functionality to the product and passes it on to the next stage. The first increment is generally known as a **core product** and is used by the user for a detailed evaluation.
- This process results in creation of a plan for the next increment.
- This plan determines the modifications (features or functions) of the product in order to accomplish user requirements.
- The iteration process, which includes the delivery of the increments to the user, continues until the software is completely developed.

## Incremental Model Strengths

- **Early feedback is generated** because implementation occurs rapidly for a small subset of the software.
- There is an **emphasis on reuse**.
- Risk of changing requirements is reduced
- In iterative model **we are building and improving the product step by step. Hence we can track the defects at early stages.**
- In iterative model **less time is spent on documenting and more time is given for designing**.

## Incremental Model Weaknesses

- **Rework may increase**
- **Requires good planning and design**
- **Requires early definition of a complete and fully functional system** to allow for the definition of increments
- **Becomes invalid when there is time constraint** on the project schedule or when the users cannot accept the phased deliverables.
- **Costly system architecture** or design issues may arise because not all requirements are gathered up front for the entire lifecycle

## Questions asked in Software Companies

- What are the benefits of prototyping?[Wipro]
- What are the disadvantages of classic life cycle model?[HCL]
- What are the merits of the incremental model? [HCL]

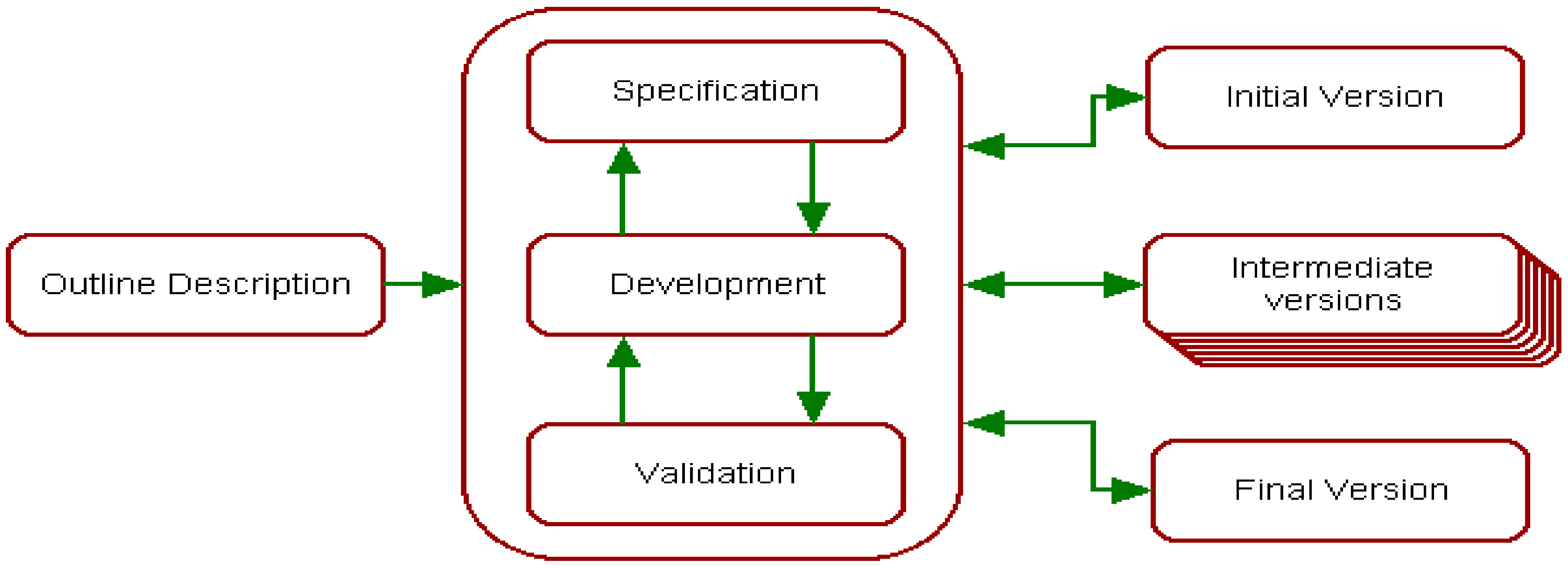
## EVOLUTIONARY PROCESS MODEL

- Evolutionary process model **resembles iterative enhancement model.**
- **The objective is to work with client and to evolve a final system from an initial outline specification.**
- **Should start with well-understood requirements.**
- These models are applied because as the **requirements often change so the end product will be unrealistic**, where a complete version is impossible, so due to tight market deadlines it is better to introduce a limited version to meet the pressure.

## EVOLUTIONARY PROCESS MODEL

- **The process is iterative** as the software engineer goes through a repetitive process of requirement until all users and stakeholders are satisfied.
- **This model differs from the iterative enhancement model** in the sense that this does not require a useable product at the end of each cycle.
- In evolutionary development, requirements are implemented by category rather than by priority.

# EVOLUTIONARY PROCESS MODEL



**Specification, development and validation activities are concurrent with strong feedback between each.**

## EVOLUTIONARY PROCESS MODEL

A software process model is a structured set of activities required to develop a software system.

The following are the generic parts:

**Specification** – this is the stage at which requirements are drawn up.

**Development** – at this stage the specification is coded into either a prototype or the finished product.

**Validation** – at this stage the client is given his or her acceptance to the software development.

**Evolution** – the client may decide to make minor or major changes or further the existing specification to improve the software being developed.

## APPLICATIONS

- For **small or medium-size** interactive systems
- This model is **useful for projects using new technology** that is not well understood.
- This is also **used for complex projects where all functionality must be delivered at one time, but the requirements are unstable or not well understood at the beginning**.

## ADVANTAGES

- **Customer involvement** in the process:
  - More likely to meet the user requirement.
- **Provides quickly an initial version of the system**
- This model is very **appropriate for research projects**.

## DISADVANTAGES

- **Not suitable for large applications**
- **User can get too involved**
- It is **difficult to measure progress and produce documentation reflecting every version of the system as it evolves.**

## Types of Evolutionary Models

Types of evolutionary models

- **Prototyping Model**
- **Spiral Model**

## PROTOTYPING MODEL

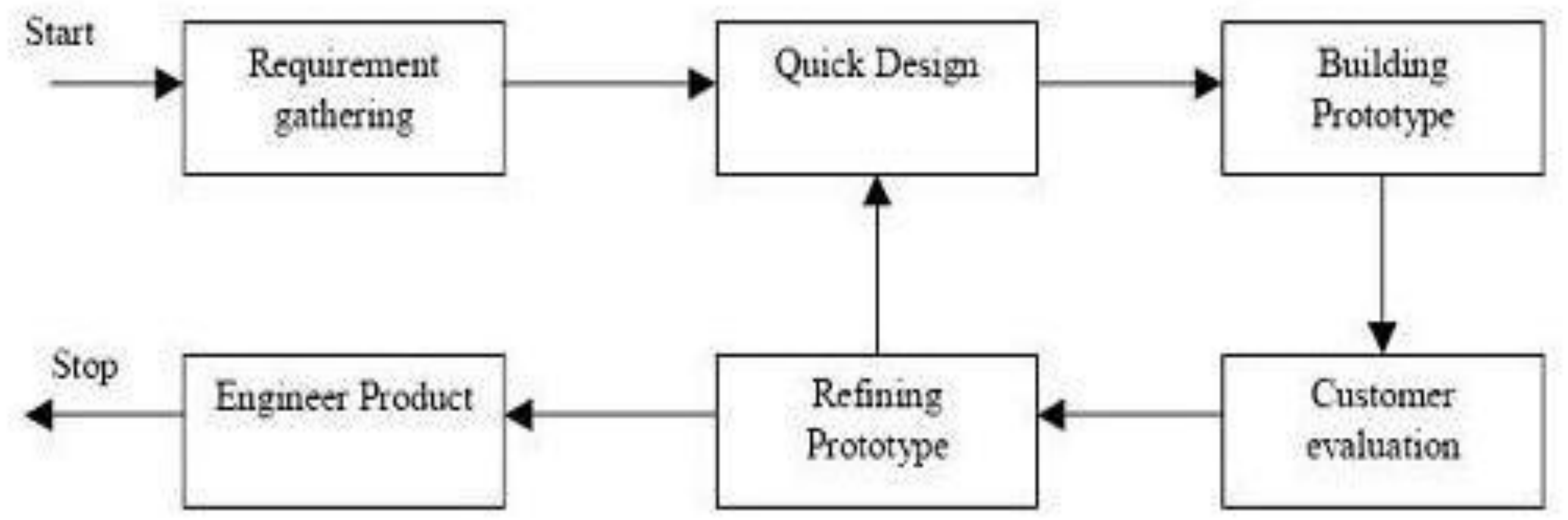
- The **prototyping model** is applied when **detailed information related to input and output requirements of the system is not available**
- This model allows the users to interact and experiment with a working model of the system known as **prototype**
- The prototype gives the user an actual feel of the system.
- For example, **ecommerce website**

## What is Prototype

- A prototype is the sample implementation of the real system.
- It shows limited and main functional capabilities of the proposed system.
- It is prepared by creating **main user interfaces without any coding.**
- **It helps the customer determine how the feature will function in the final software.**

## PROTOTYPING

Figure Illustrates the steps carried out in the prototyping model.



Prototyping Model

## PROTOTYPING

1. **Requirements gathering and analysis:** A prototyping model begins with requirements analysis and the requirements of the system are defined in detail.
2. **Quick design:** When requirements are known, a preliminary design or quick design for the system is created. It is not a detailed design and includes only the important aspects of the system.
3. **Build prototype:** Information gathered from quick design is modified to form the first prototype, which represents the working model of the required system.

4. **User evaluation:** Next, the proposed system is presented to the user for thorough evaluation of the prototype. Comments and suggestions are collected from the users and provided to the developer.
5. **Refining prototype:** Once the user evaluates the prototype and if he is not satisfied, the current prototype is refined according to the requirements. Once the user is satisfied with the developed prototype, a final system is developed on the basis of the final prototype.
6. **Engineer product:** Once the requirements are completely met, the user accepts the final prototype. The final system is evaluated thoroughly followed by the routine maintenance.

## ADVANTAGES OF PROTOTYPING

- When prototype is shown to the user, he gets a proper clarity and 'feel' of the functionality of the software and he can suggest changes and modifications.
- This type of approach of developing the software is used for non-IT-literate people.
- **When client is not confident about the developer's capabilities**, he asks for a small prototype to be built. Based on this model, he judges capabilities of developer.

## LIMITATION OF PROTOTYPING

- If the user is not satisfied by the developed prototype, then a new prototype is developed. This process goes on until a perfect prototype is developed.
- Thus, **this model is time consuming and expensive.**

## When to use Prototype model

- Prototype model should be used **when the desired system needs to have a lot of interaction with the end users**. Example online systems, web interfaces etc.
- Prototyping ensures that the end users constantly work with the system and provide a feedback which is incorporated in the prototype to result in a useable system.

## THE SPIRAL MODEL

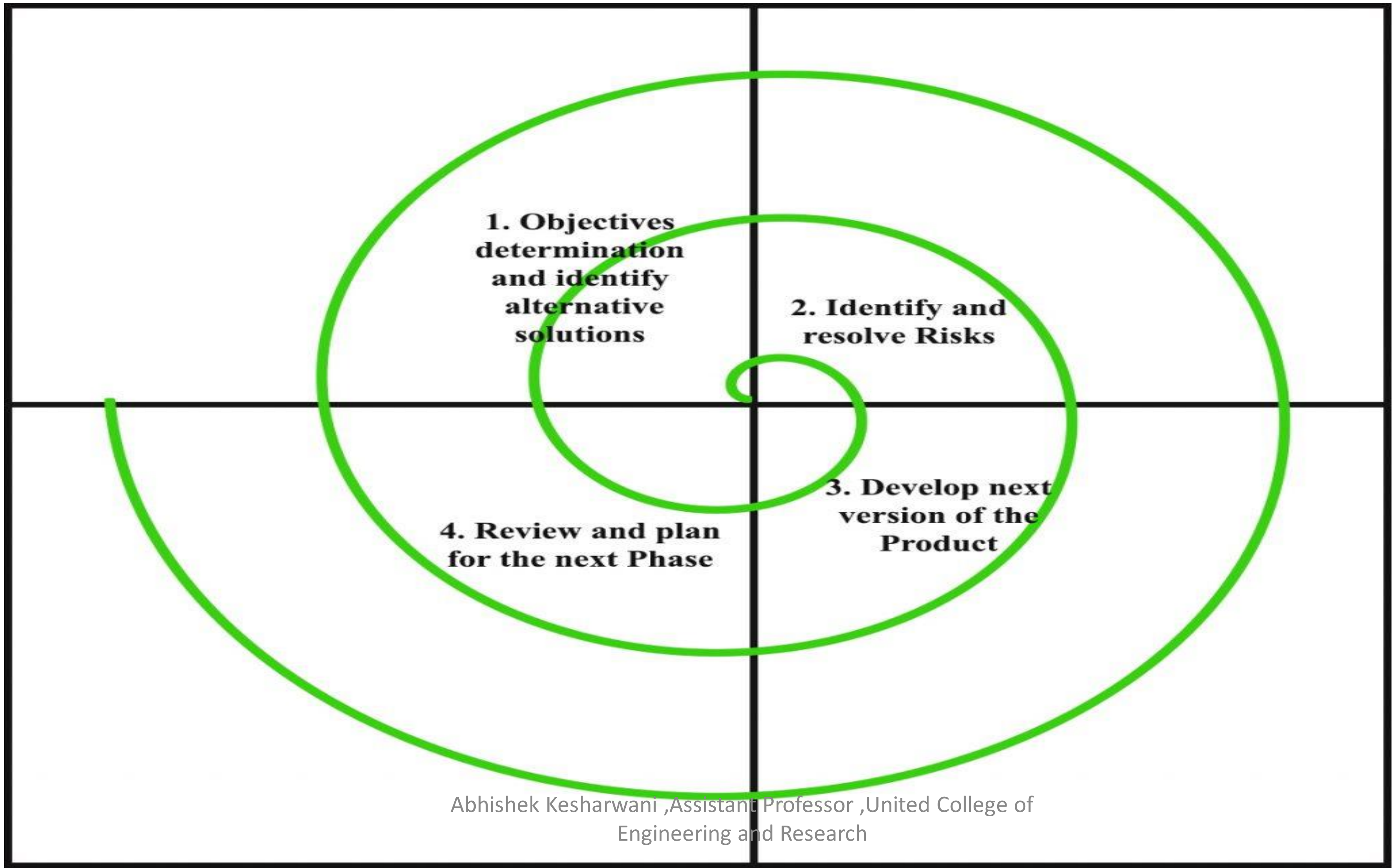
This model of development combines the features of the prototyping model and the systems development life cycle (SDLC) **with very high emphasis on risk analysis.**

- It allows for incremental releases of the product, or incremental refinement through each iteration around the spiral.

Process is represented as a *spiral* rather than as a sequence of activities with backtracking.

- Each loop = One Iteration = A process phase.
- As loops move away from center → Time and Cost increase

# Pictorial Representation of SDLC Spiral Model


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

## Phases of Spiral Model

- **Planning Phase:** Requirements are gathered during the planning phase.
- **Risk Analysis:** In the risk analysis phase, a process is undertaken to identify risk and alternate solutions.  A prototype is produced at the end of the risk analysis phase. If any risk is found during the risk analysis then alternate solutions are suggested and implemented.
- **Engineering Phase:** In this phase software is developed, along with testing at the end of the phase. Hence in this phase the development and testing is done.
- **Evaluation phase:** This phase allows the customer to evaluate the output of the project to date before the project continues to the next spiral.

## Advantages

- High amount of risk analysis hence, avoidance of Risk is enhanced.
- **The spiral model is favored for large, expensive, and complicated projects.**
- Development can be divided into smaller parts and more risky parts can be developed earlier which helps better risk management.

## Disadvantages

- Can be a costly model to use.
- Risk analysis is important phase, hence requires expert people.
- Project’s success is highly dependent on the risk analysis phase.
- Not suitable for small or low risk projects and could be expensive for small projects.
- Spiral may go indefinitely(infinitely).