CHITKARA UNIVERSITY
HIMACHAL PRADESH - NAAC Accredited

Object Oriented Software Engineering

LECTURE 3

CHITKARA UNIVERSITY
HIMACHAL PRADESH - NAAC Accredited

Today's Outline:

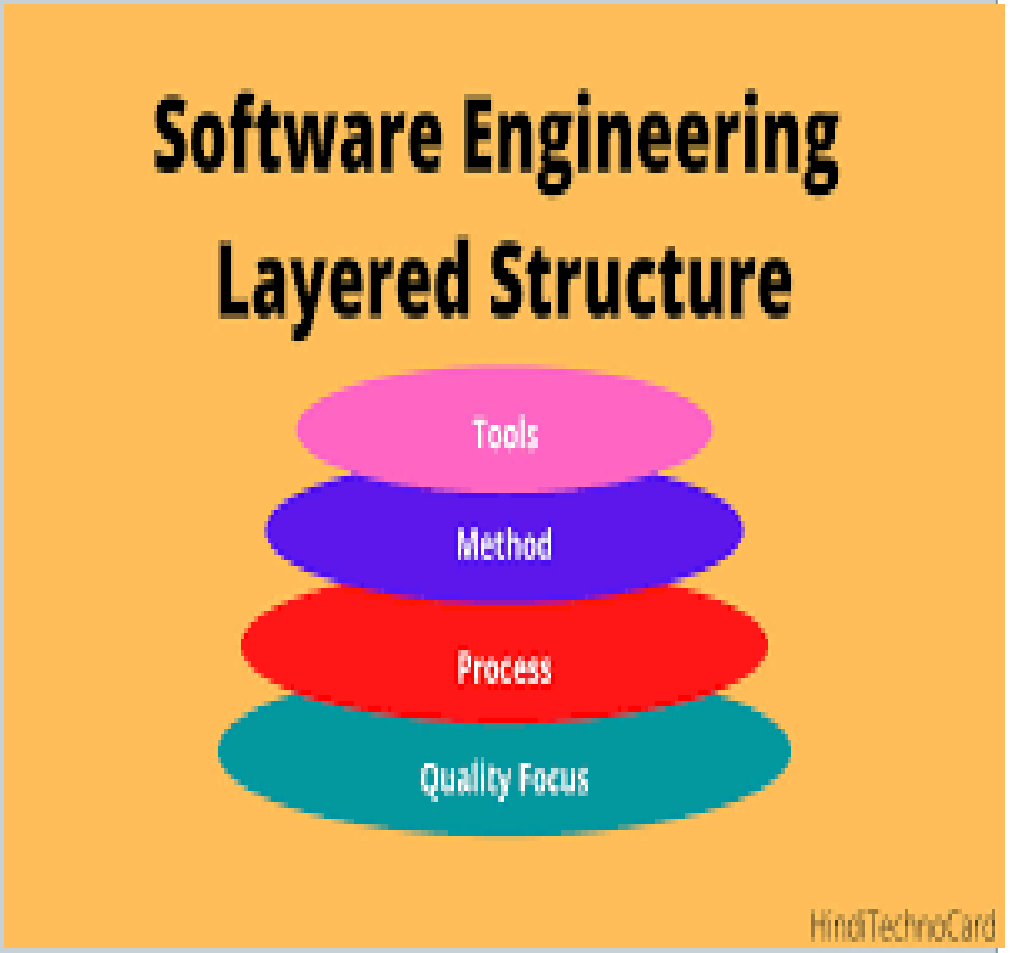
The Software Process

Software Engineering–Layered Technology

Process Models: The Waterfall Model

Software Engineering: A Layered Approach

- Software engineering teams are required to analyze user needs and then go on to design, implement and test their end product to make sure it satisfies those needs through the use of programming languages.



A Layered Technology

- **The layered technology consists of:**

  1. Quality focus
  The characteristics of good quality software are:
- Correctness of the functions required to be performed by the software.
- Maintainability of the software
- Integrity i.e. providing security so that the unauthorized user cannot access information or data.
- Usability i.e. the efforts required to use or operate the software.
- **2. Process:** It is the base layer or foundation layer for the software engineering.
- The software process is the key to keep all levels together.
- It defines a framework that includes different activities and tasks.
- In short, it covers all activities, actions and tasks required to be carried out for software development.

A Layered Technology

3. Methods: The method provides the answers of all 'how-to' that are asked during the process.
It provides the technical way to implement the software.
It includes collection of tasks starting from communication, requirement analysis, analysis and design modelling, program construction, testing and support.
4. Tools: The software engineering tool is an automated support for the software development.
The tools are integrated i.e the information created by one tool can be used by the other tool.
For example: The Microsoft publisher can be used as a web designing tool.

CHITKARA UNIVERSITY
HIMACHAL PRADESH - NAAC Accredited

Process Models

- **What is a software process model?**
- A software process model is an abstraction of the software development process. The models specify the stages and order of a process. So, think of this as a representation of the **order of activities** of the process and the **sequence** in which they are performed.

Process Models

A model will define the following:

- The tasks to be performed
- The input and output of each task
- The pre and post conditions for each task
- The flow and sequence of each task

The goal of a software process model is to provide guidance for controlling and coordinating the tasks to achieve the end product and objectives as effectively as possible.

CHITKARA UNIVERSITY
HIMACHAL PRADESH - NAAC Accredited

What is a Software Process Model?

- Software Processes is a coherent set of activities for specifying, designing, implementing and testing software systems. A software process model is an abstract representation of a process that presents a description of a process from some particular perspective. There are many different software processes but all involve:
- Specification – defining what the system should do;
- Design and implementation – defining the organization of the system and implementing the system;
- Validation – checking that it does what the customer wants;
- Evolution – changing the system in response to changing customer needs.

CHITKARA UNIVERSITY
HIMACHAL PRADESH - NAAC Accredited

Process Models

- There are many kinds of process models for meeting different requirements. We refer to these as **SDLC models** (Software Development Life Cycle models). The most popular and important SDLC models are as follows:
- Waterfall model
- V model
- Incremental model
- RAD model
- Agile model
- Iterative model
- Prototype model
- Spiral model

Factors in choosing a software process

- Choosing the right software process model for your project can be difficult. If you know your requirements well, it will be easier to select a model that best matches your needs. You need to keep the following factors in mind when selecting your software process model:
- **Project requirements**
- Before you choose a model, take some time to go through the project requirements and clarify them alongside your organization's or team's expectations. Will the user need to specify requirements in detail after each iterative session? Will the requirements *change* during the development process?
- **Project size**
- Consider the size of the project you will be working on. Larger projects mean bigger teams, so you'll need more extensive and elaborate project management plans.

Factors in choosing a software process

- **Project complexity**
- Complex projects may not have clear requirements. The requirements may change often, and the cost of delay is high. Ask yourself if the project requires constant monitoring or feedback from the client.
- **Cost of delay**
- Is the project highly time-bound with a huge cost of delay, or are the timelines flexible?
- **Customer involvement**
- Do you need to consult the customers during the process? Does the user need to participate in all phases?
- **Familiarity with technology**
- This involves the developers' knowledge and experience with the project domain, software tools, language, and methods needed for development.
- **Project resources**
- This involves the amount and availability of funds, staff, and other resources.

CHITKARA UNIVERSITY
HIMACHAL PRADESH - NAAC Accredited

Software Development Life Cycle (SDLC)

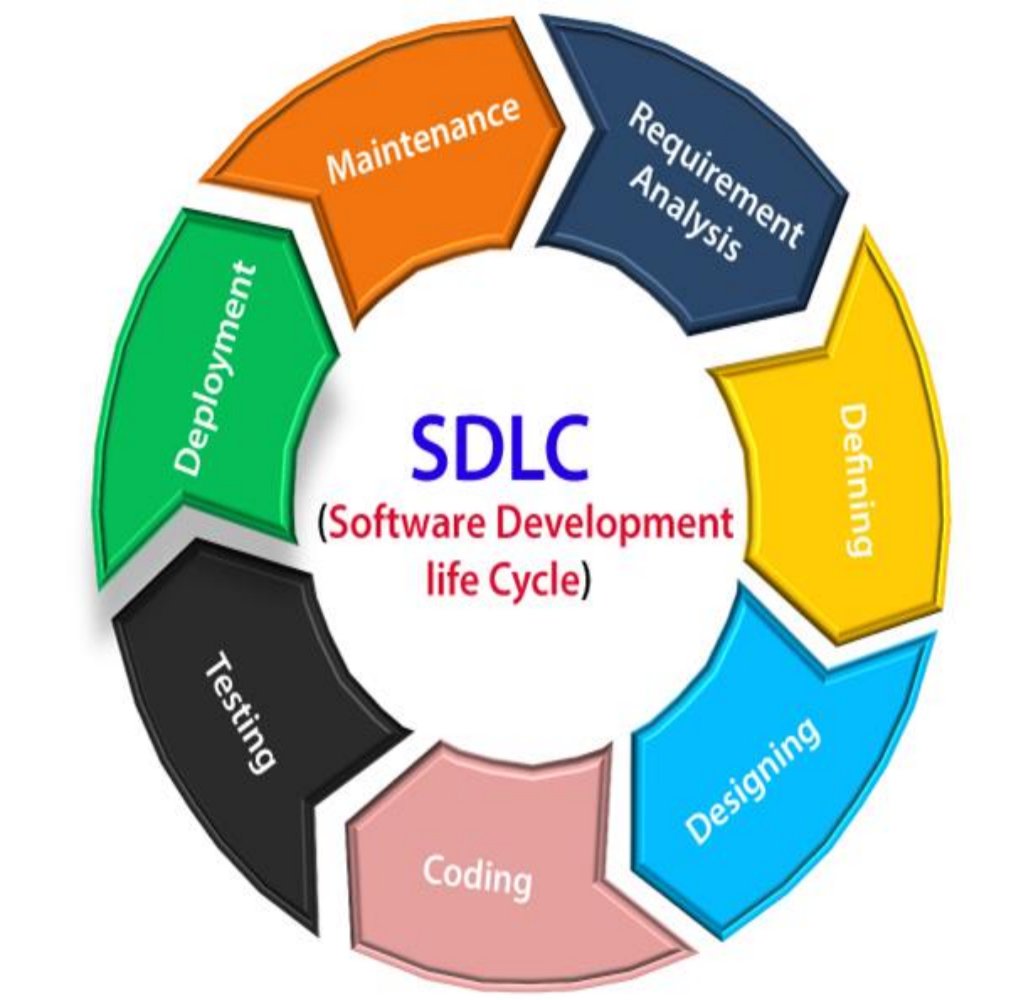
- A software life cycle model (also termed process model) is a pictorial and diagrammatic representation of the software life cycle. A life cycle model represents all the methods required to make a software product transit through its life cycle stages. It also captures the structure in which these methods 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

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



CHITKARA UNIVERSITY
HIMACHAL PRADESH - NAAC Accredited

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.
- 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.

The stages of SDLC are as follows:

- **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.

The stages of SDLC are as follows:

- **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.

The stages of SDLC are as follows:

- **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.

The stages of SDLC are as follows:

- **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.

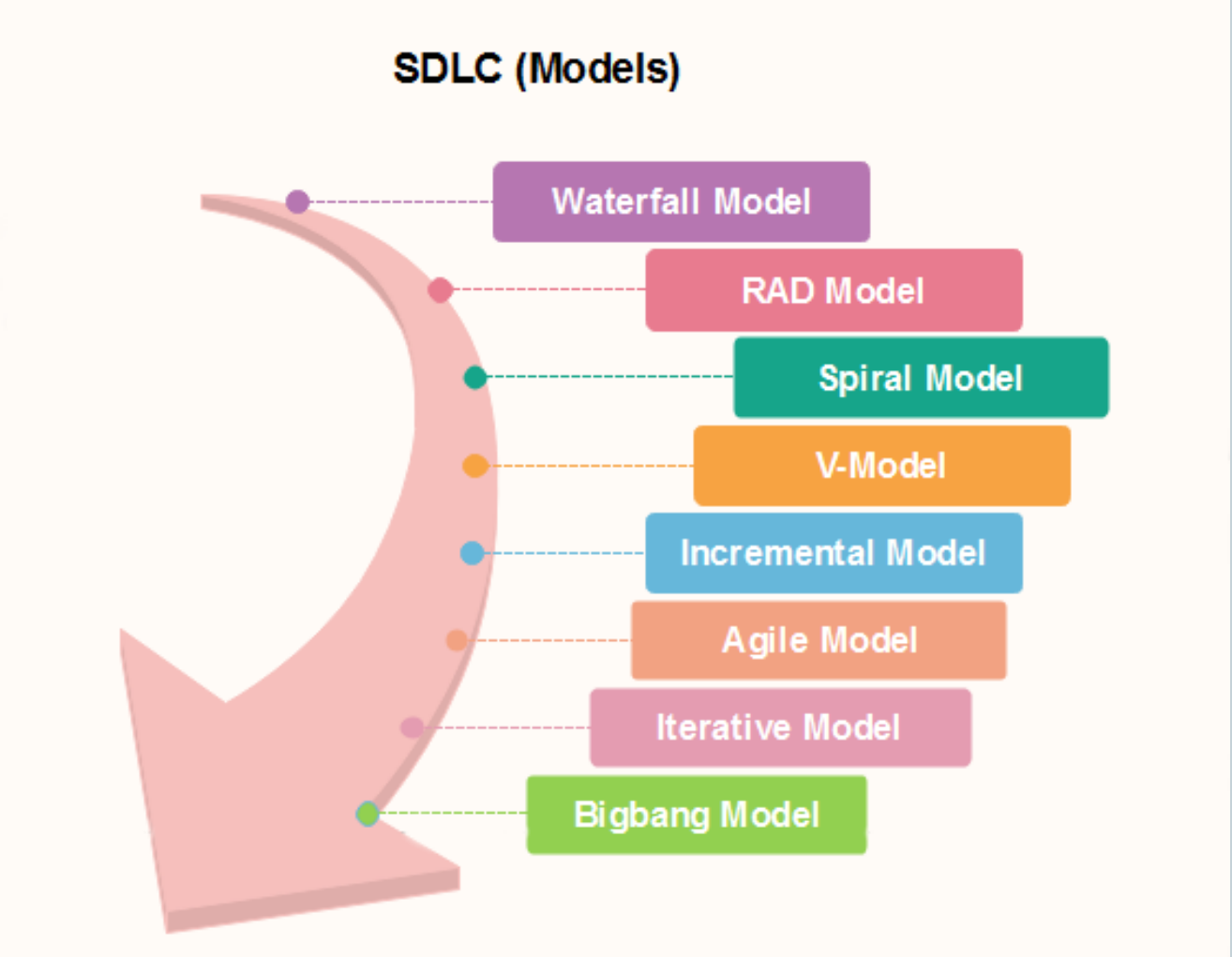
The stages of SDLC are as follows:

- **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.
- This procedure where the care is taken for the developed product is known as maintenance.

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


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

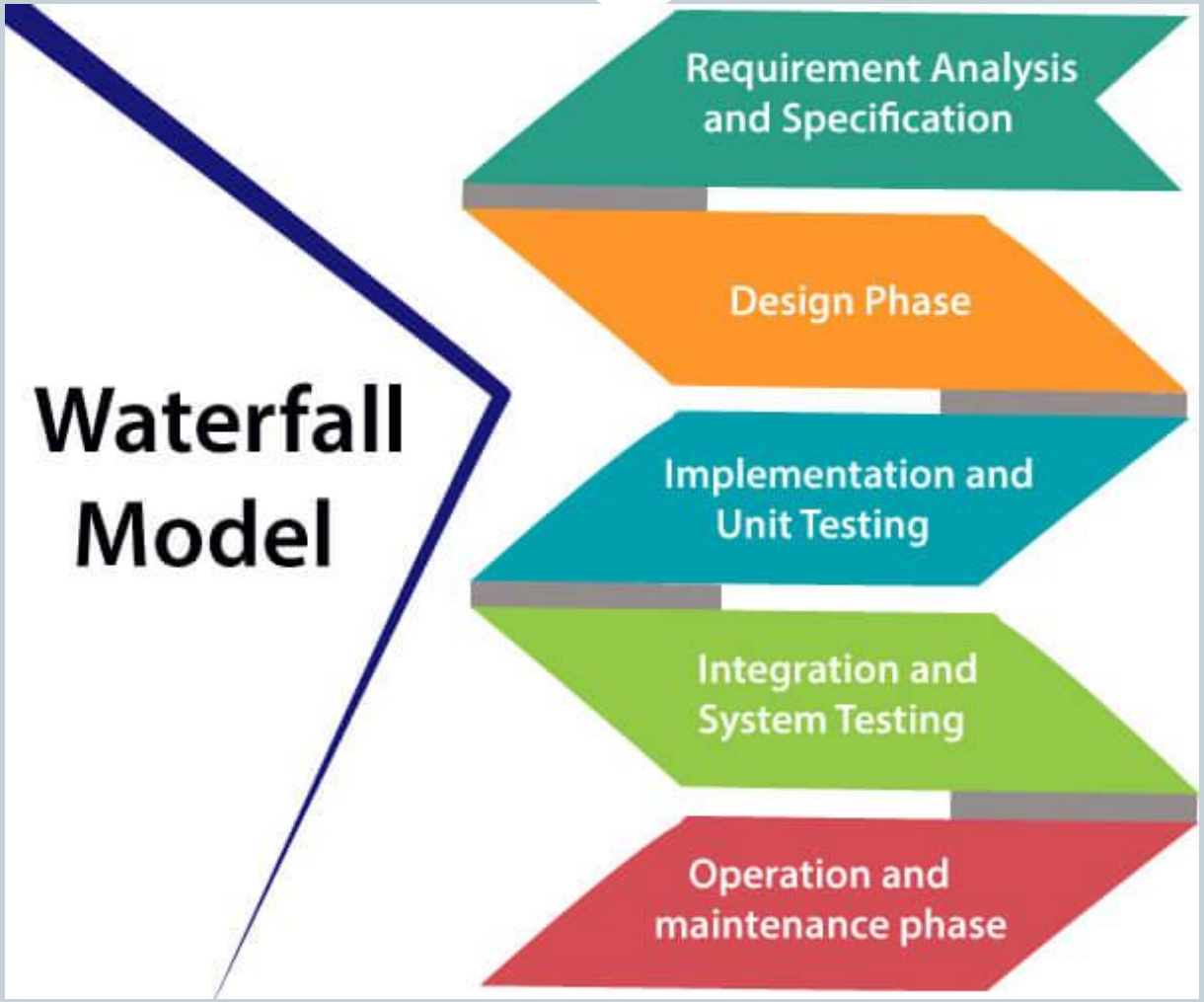
CHITKARA UNIVERSITY
HIMACHAL PRADESH - NAAC Accredited

The Waterfall Model

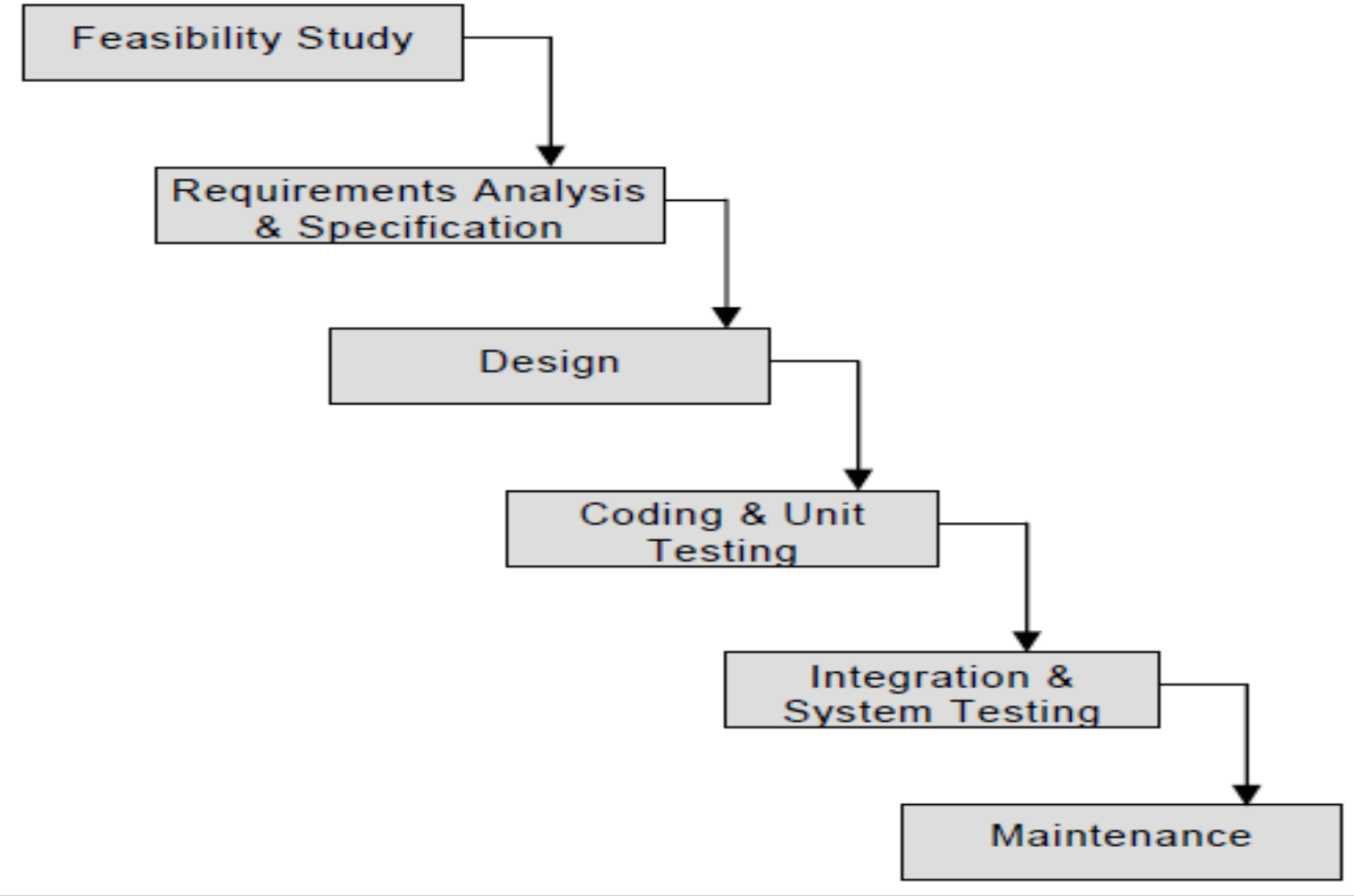
Winston Royce introduced the Waterfall Model in 1970.This model has **five** phases: ***Requirements analysis and specification, design, implementation, and unit testing, integration and system testing, and operation and maintenance.***

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.

The Waterfall Model



Classical Waterfall Model:



Classical Waterfall Model:

- *Classical Waterfall Model*:
- The waterfall Model illustrates the software development process in a linear sequential flow. This means that any phase in the development process begins only if the previous phase is complete. In this waterfall model, the phases do not overlap.

Classical Waterfall Model:

- *Classical Waterfall Model*:

Pros:

- This model is very simple and is easy to understand.
- Phases in this model are processed one at a time.
- Each stage in the model is clearly defined.
- This model has very clear and well understood milestones.
- Process, actions and results are very well documented.
- Reinforces good habits: define-before- design, design-before-code.
- This model works well for smaller projects and projects where requirements are well understood.

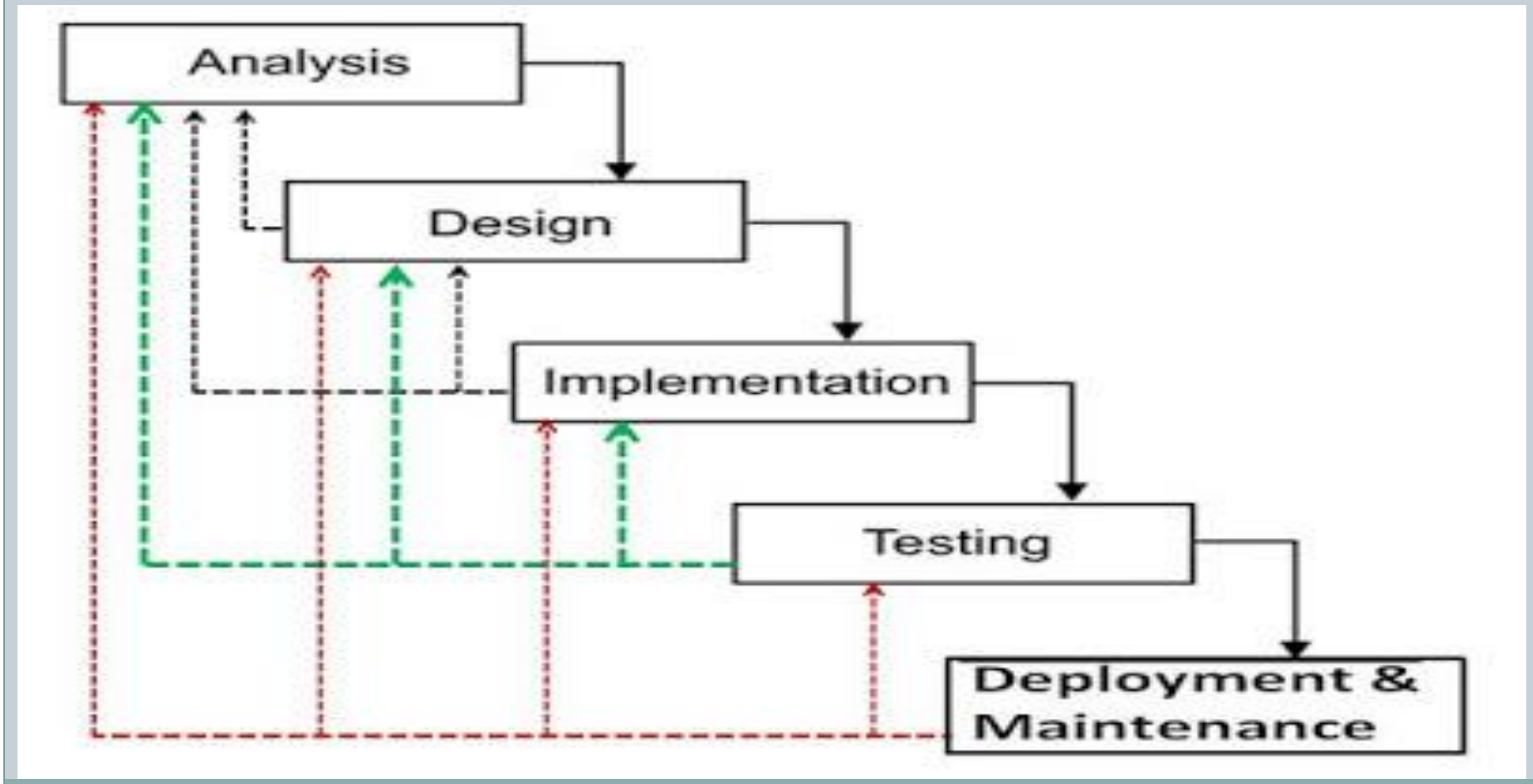
Classical Waterfall Model:

Cons:

- **No feedback path:** In classical waterfall model evolution of a software from one phase to another phase is like a waterfall. It assumes that no error is ever committed by developers during any phases. Therefore, it does not incorporate any mechanism for error correction.
- **Difficult to accommodate change requests:** This model assumes that all the customer requirements can be completely and correctly defined at the beginning of the project, but actually customers' requirements keep on changing with time. It is difficult to accommodate any change requests after the requirements specification phase is complete.
- **No overlapping of phases:** This model recommends that new phase can start only after the completion of the previous phase. But in real projects, this can't be maintained. To increase the efficiency and reduce the cost, phases may overlap.

Iterative Waterfall Model:

- *Iterative Waterfall Model*:



Iterative Waterfall Model:

- *Iterative Waterfall Model*:
- The iterative waterfall model provides feedback paths from every phase to its preceding phases, which is the main difference from the classical waterfall model.
- When errors are detected at some later phase, these feedback paths allow correcting errors committed by programmers during some phase. The feedback paths allow the phase to be reworked in which errors are committed and these changes are reflected in the later phases. But, there is no feedback path to the stage – feasibility study, because once a project has been taken, does not give up the project easily.
  It is good to detect errors in the same phase in which they are committed. It reduces the effort and time required to correct the errors.

CHITKARA UNIVERSITY
HIMACHAL PRADESH - NAAC Accredited

Iterative Waterfall Model:

- *Iterative Waterfall Model*:

Pros:

- **Feedback Path:** In the classical waterfall model, there are no feedback paths, so there is no mechanism for error correction. But in iterative waterfall model feedback path from one phase to its preceding phase allows correcting the errors that are committed and these changes are reflected in the later phases.
- **Simple:** Iterative waterfall model is very simple to understand and use. That's why it is one of the most widely used software development models.

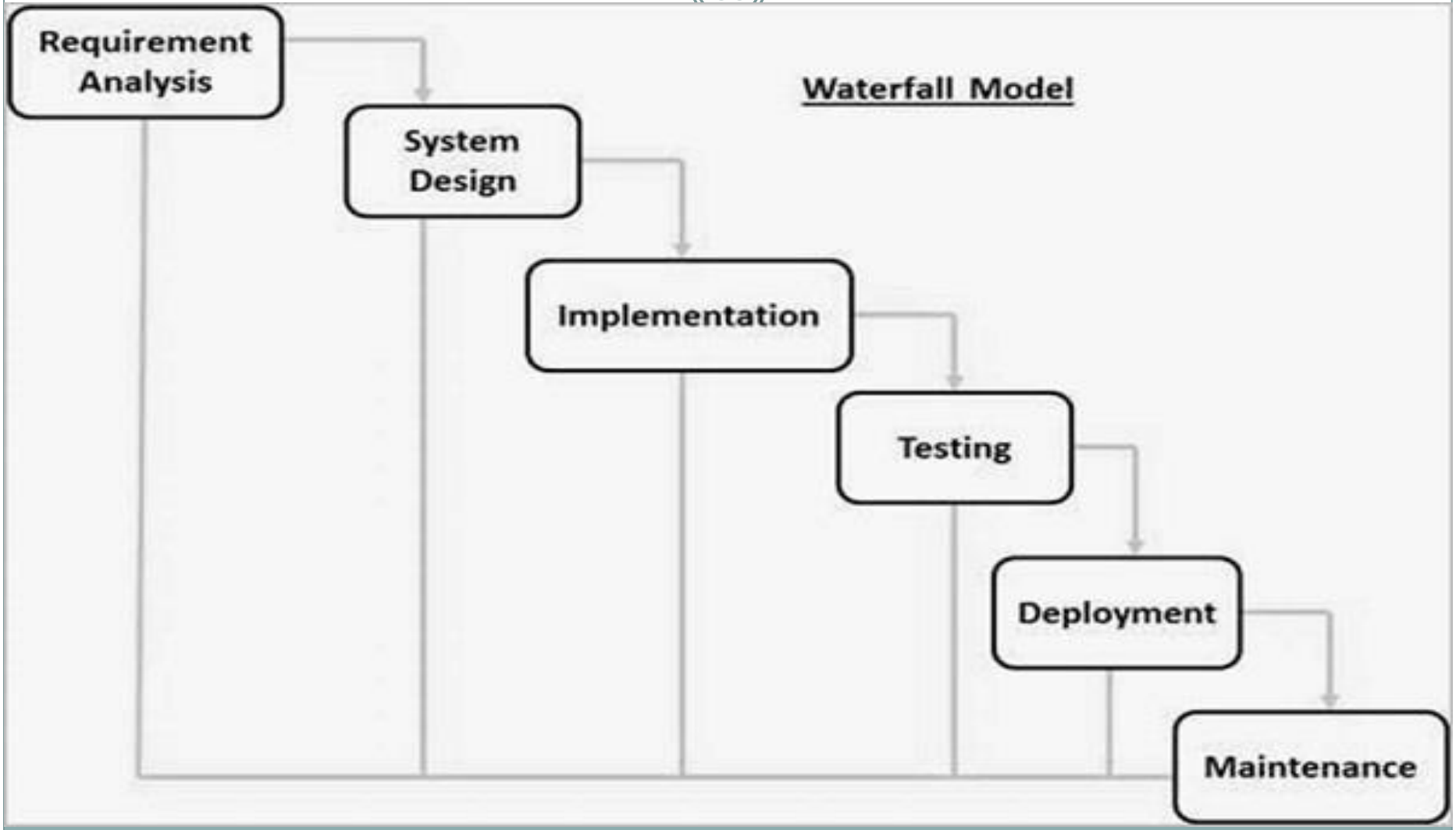
Iterative Waterfall Model:

- *Iterative Waterfall Model*:

Cons:

- **Difficult to incorporate change requests:** The major drawback of the iterative waterfall model is that all the requirements must be clearly stated before starting of the development phase. Customer may change requirements after some time but the iterative waterfall model does not leave any scope to incorporate change requests that are made after development phase starts.
- **Incremental delivery not supported:** In the iterative waterfall model, the full software is completely developed and tested before delivery to the customer. There is no scope for any intermediate delivery. So, customers have to wait long for getting the software.
- **Overlapping of phases not supported:** Iterative waterfall model assumes that one phase can start after completion of the previous phase, But in real projects, phases may overlap to reduce the effort and time needed to complete the project.
- **Risk handling not supported:** Projects may suffer from various types of risks. But, Iterative waterfall model has no mechanism for risk handling.
- **Limited customer interactions:** Customer interaction occurs at the start of the project at the time of requirement gathering and at project completion at the time of software delivery. These fewer interactions with the customers may lead to many problems as the finally developed software may differ from the customers' actual requirements.

The Waterfall Model



The Waterfall Model

- **1. Requirements analysis and specification phase:**

The aim of this phase is to understand the exact requirements of the customer and to document them properly. Both the customer and the software developer work together so as to document all the functions, performance, and interfacing requirement of the software. It describes the "what" of the system to be produced and not "how."In this phase, a large document called **Software Requirement Specification (SRS)** document is created which contained a detailed description of what the system will do in the common language.

The Waterfall Model

2. Design Phase: This phase aims to transform the requirements gathered in the SRS into a suitable form which permits further coding in a programming language. It defines the overall software architecture together with high level and detailed design. All this work is documented as a Software Design Document (SDD).

The Waterfall Model

- **3. Implementation and unit testing:** During this phase, design is implemented. If the SDD is complete, the implementation or coding phase proceeds smoothly, because all the information needed by software developers is contained in the SDD.
- During testing, the code is thoroughly examined and modified. Small modules are tested in isolation initially. After that these modules are tested by writing some overhead code to check the interaction between these modules and the flow of intermediate output.

CHITKARA UNIVERSITY
HIMACHAL PRADESH - NAAC Accredited

The Waterfall Model

- **4. Integration and System Testing:** This phase is highly crucial as the quality of the end product is determined by the effectiveness of the testing carried out. The better output will lead to satisfied customers, lower maintenance costs, and accurate results. Unit testing determines the efficiency of individual modules. However, in this phase, the modules are tested for their interactions with each other and with the system.
- **5. Operation and maintenance phase:** Maintenance is the task performed by every user once the software has been delivered to the customer, installed, and operational.

CHITKARA UNIVERSITY
HIMACHAL PRADESH - NAAC Accredited

When to use SDLC Waterfall Model?

Some Circumstances where the use of the Waterfall model is most suited are:

- When the requirements are constant and not changed regularly.
- A project is short
- The situation is calm
- Requirement is clear
- Where the tools and technology used is consistent and is not changing
- When resources are well prepared and are available to use.

Advantages of Waterfall model

This model is simple to implement also the number of resources that are required for it is minimal.

- The requirements are simple and explicitly declared; they remain unchanged during the entire project development.
- The start and end points for each phase is fixed, which makes it easy to cover progress.
- The release date for the complete product, as well as its final cost, can be determined before development.
- It gives easy to control and clarity for the customer due to a strict reporting system.

CHITKARA UNIVERSITY
HIMACHAL PRADESH - NAAC Accredited

Disadvantages of Waterfall model

- In this model, the risk factor is higher, so this model is not suitable for more significant and complex projects.
- This model cannot accept the changes in requirements during development.
- It becomes tough to go back to the phase. For example, if the application has now shifted to the coding phase, and there is a change in requirement, It becomes tough to go back and change it.
- Since the testing done at a later stage, it does not allow identifying the challenges and risks in the earlier phase, so the risk reduction strategy is difficult to prepare.

THANK YOU