Software Life Cycle Models

Outline

- **What is software life cycle model**
- SDLC
- Why SDLC?
- Build and Fix Model

SOFTWARE LIFE CYCLE MODELS

- It is often called a **SDLC.**
- Ultimate objective of SE is **“to produce good quality maintainable software within reasonable time frame, and at affordable cost”.**
- Must have a matured process.
- A key component of any software development process is the **life cycle model** on which the process is based.
- Life cycle of the software starts from concept exploration and ends at the retirement of the software.

SDLC

There are following six phases in every Software development life cycle model:

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

1. Requirement gathering and analysis

Business requirements are gathered in this phase. This phase is the main focus of the project managers and stake holders. Meetings with managers, stake holders and users are held in order to determine the requirements like; Who is going to use the system? How will they use the system? What data should be input into the system? What data should be output by the system? These are general questions that get answered during a requirements gathering phase. After requirement gathering these requirements are analyzed for their validity and the possibility of incorporating the requirements in the system to be development is also studied.

2. Design

In this phase the system and software design is prepared from the requirement specifications which were studied in the first phase. System Design helps in specifying hardware and system requirements and also helps in defining overall system architecture. The system design specifications serve as input for the next phase of the model.

3. Implementation / Coding

On receiving system design documents, the work is divided in modules/units and actual coding is started. Since, in this phase the code is produced so it is the main focus for the developer. This is the longest phase of the software development life cycle.

4. Testing

After the code is developed it is tested against the requirements to make sure that the product is actually solving the needs addressed and gathered during the requirements phase. During this phase unit testing, integration testing, system testing, acceptance testing are done.

5. Deployment

After successful testing the product is delivered / deployed to the customer for their use.

6. Maintenance

Once when the customers starts using the developed system then the actual problems comes up and needs to be solved from time to time. This process where the care is taken for the developed product is known as maintenance.

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**

Life cycle Models

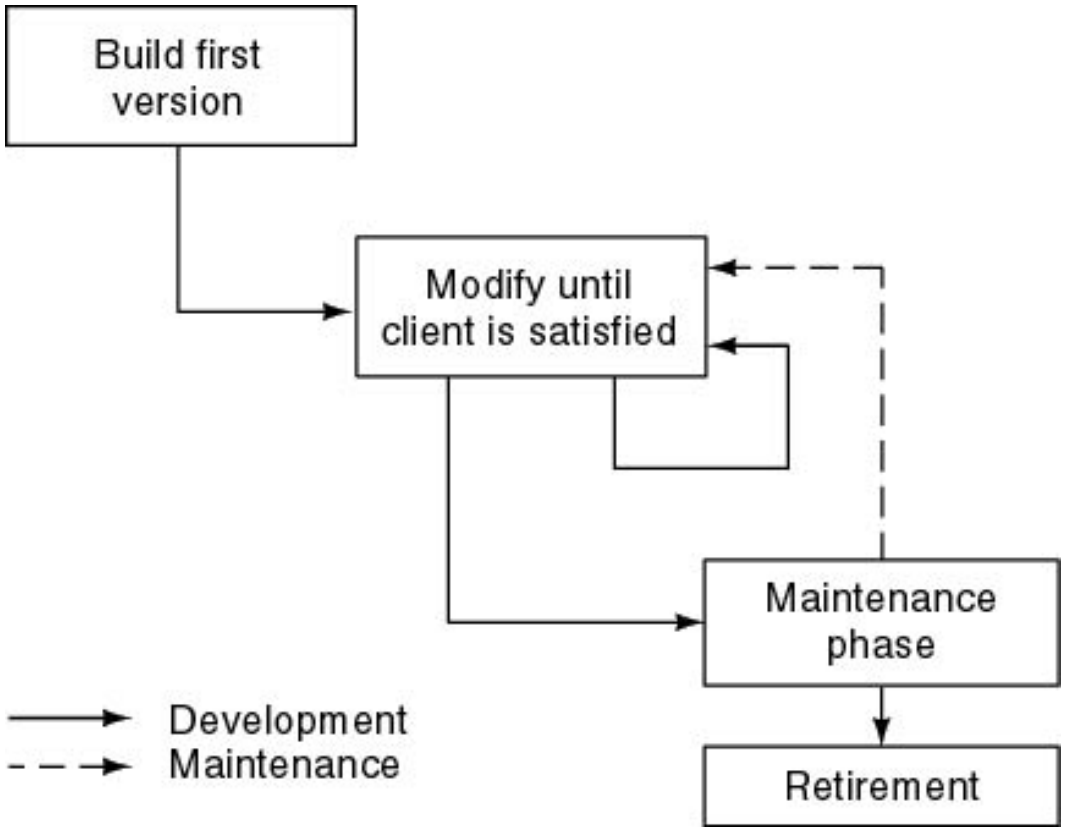
- Different software life cycle models have been proposed so far. Each of them has some advantages as well as some disadvantages.

- Several different process models vary mainly in the frequency, application and implementation of the above techniques, for example, different process models use different analysis techniques, other models attempt to implement the solution to a problem in one big-bang approach, while others adopt an iterative approach whereby successively larger and more complete versions of the software are built with each iteration of the process model.

- A few important and commonly used life cycle models are as follows:

  1. Build and Fix Model
  2. Classical Waterfall Model
  3. Iterative Model
  4. Prototyping Model
  5. Evolutionary Model
  6. Spiral Model
  7. Rapid Application Development model (RAD)

Build and Fix Model



Notes

- 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.

Notes

- 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.**

Outline

- **Waterfall Model**
- Applications
- Advantages
- Disadvantages
- Problems

What is Waterfall Model?

- First Process Model
- Oldest model for software engineering, hence called as **Classical Life Cycle Model.**
- It is very simple to understand and use.
- It is also referred to as a linear-sequential life cycle model, because its software development process follows a linear sequential flow.

The Waterfall Model

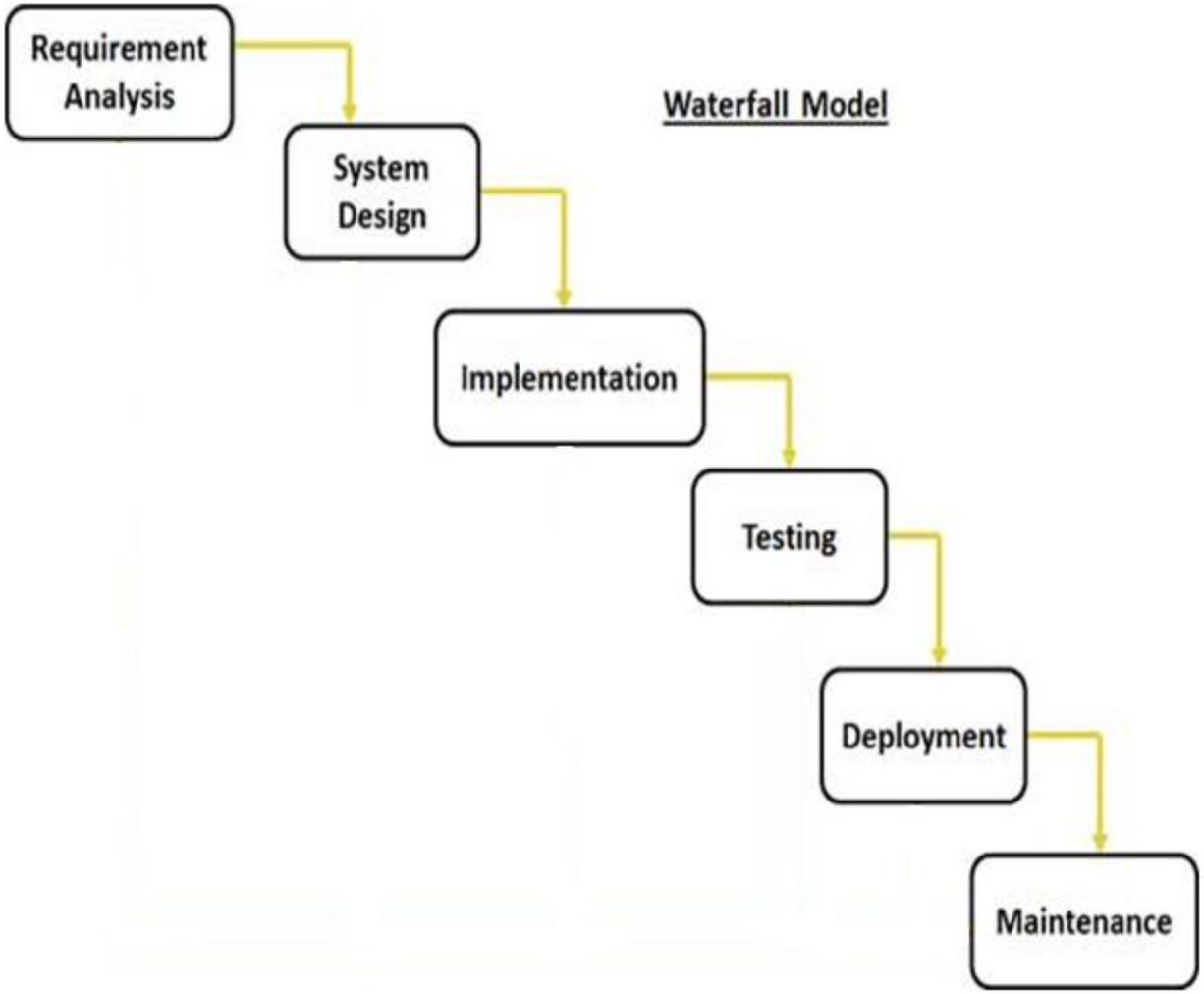
Contd....

This model is easy to understand and reinforces the notion of "define before design" and "design before code".

The model expects complete & accurate requirements early in the process, which is unrealistic

The Waterfall Model

Contd....



Waterfall Model

This model is named **"waterfall model"** because its diagrammatic representation resembles a cascade of waterfalls.

The Waterfall Model

Contd....

The sequential phases in Waterfall model are:

- **Requirement analysis:** All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification doc.
- **System Design:** The requirement specifications from first phase are studied in this phase and system design is prepared. System Design helps in specifying hardware and system requirements and also helps in defining overall system architecture.
- **Implementation:** With inputs from 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.
- **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.

Outline

- Waterfall Model
- **Applications**
- Advantages
- Disadvantages
- Problems

Waterfall Application

Some situations **where the use of Waterfall model is most appropriate are:**

- The project is short.
- 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.
- Porting an existing product to a new platform.
- Well suited for projects where requirements can be understood easily and technology decisions are easy i.e. for familiar type of projects it still may be the most optimum

Outline

- Waterfall Model
- Applications
- **Advantages**
- Disadvantages
- Problems

Waterfall Advantages

- This model is **simple and easy to understand** and use.
- **Being a linear model**, it is very simple to implement. The **amount of resources required to implement this model are minimal.**
- Waterfall model **works well for smaller projects** where requirements are very well understood.
- **Documentation is produced at every stage of the software's development**. This makes understanding the product designing procedure, simpler.
- After every major stage of software coding, testing is done to check the correct running of the code.
- Sets requirements stability

Outline

- Waterfall Model
- Applications
- Advantages
- **Disadvantages**
- Problems

Waterfall Disadvantages

- Assumes that requirements can be specified and frozen early
- Not suitable for the projects where requirements are at a moderate to high risk of changing.
- Follows the “big bang” approach – **all or nothing delivery.**
- High amounts of risk and uncertainty.
- Very document oriented, requiring docs at the end of each phase
- You cannot go back a step, if the design phase has gone wrong, things can get very complicated in the implementation phase.
- Not a good model for long, complex and object-oriented projects.

Outline

- Waterfall Model
- Applications
- Advantages
- Disadvantages
- **Problems**

Waterfall Problems

- It **doesn't scale up** well for large projects.
- This model is **not suitable for accommodating any change**.
- A working model is not available until late in the project time plan
- Real projects rarely follow the sequential flow since they are always iterative
- It is difficult to define all requirements at the beginning of the project.
- **Once the product is ready then only it can be demoted to the end users.** Once the product is developed and if any failure occurs then the cost of fixing such issues are very high, because we need to update everywhere from document till the logic.

Outline

- **Iterative Model**
- Strengths
- Weaknesses
- When to use Iterative Model
- Waterfall vs. Iterative Model

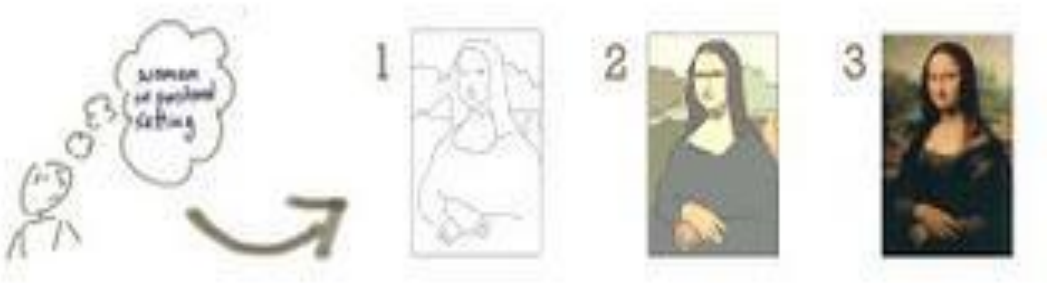
Iterative Model

- Counters the "**all or nothing**" drawback of the waterfall model
- Develop and deliver software in increments(small pieces)
- Feedback from one iteration is used in the future iterations
- It **does not attempt to start with a full specification of requirements**.
- The **basic idea** of this model is **to start the process with requirements and iteratively enhance the requirements until the final software is implemented**.

Iterative Model

Contd..

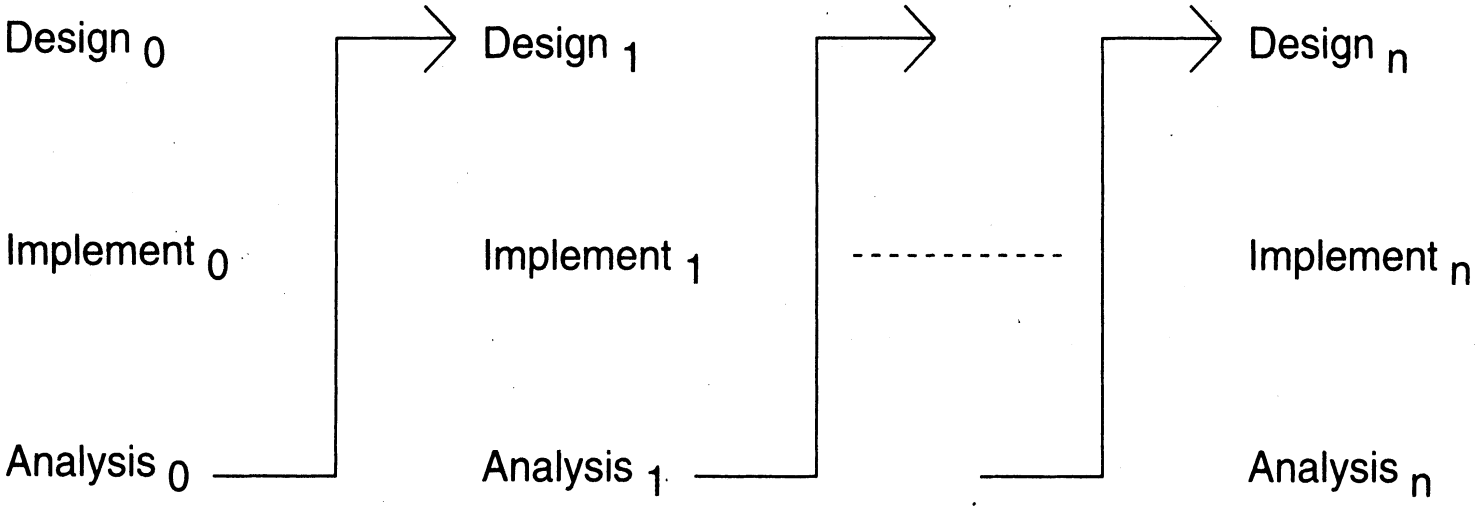
As shown in the image below, in the first iteration the **whole painting is sketched roughly**, then in the second iteration **colors are filled** and in the third iteration **finishing is done**. Hence, in iterative model the **whole product is developed step by step**.



Iterative Model

Contd..

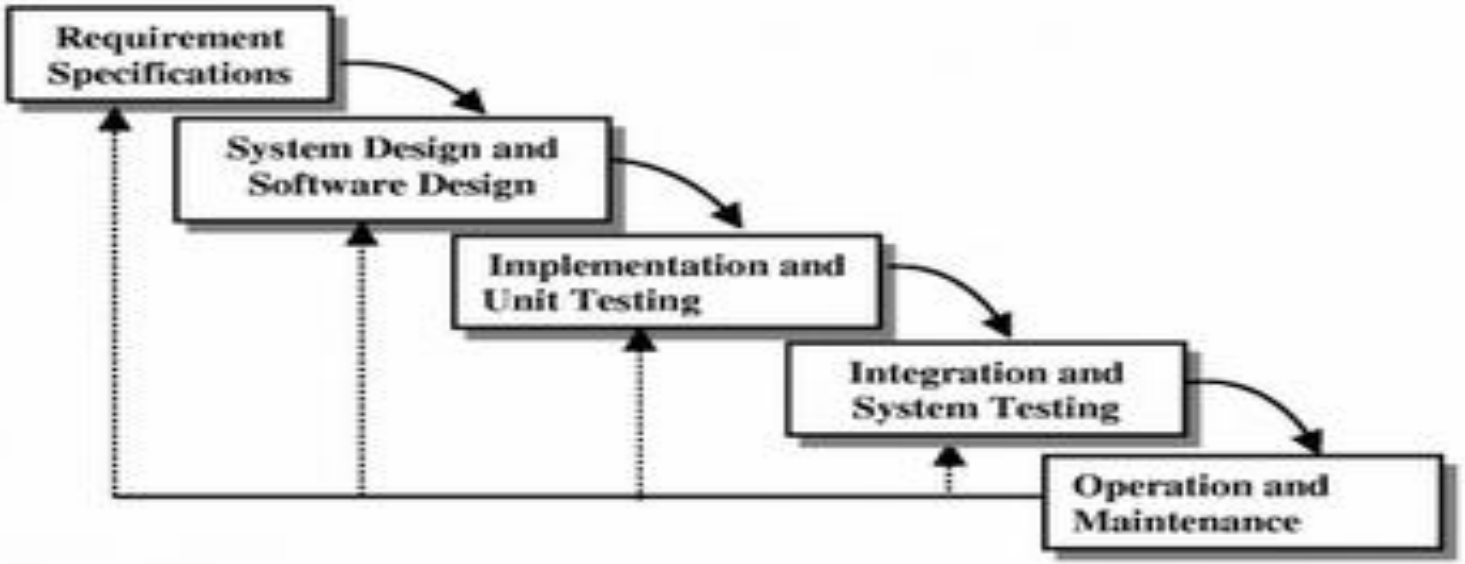
In the diagram below, when we work **iteratively we create rough product or product piece in one iteration, then review it and improve it in next iteration and so on until it's finished.**



Iterative Enhancement Model

Contd..

Can be viewed as a sequence of waterfalls



Outline

- Iterative Model
- **Strengths**
- Weaknesses
- When to use Iterative Model
- Waterfall vs. Iterative Model

Incremental Model Strengths

- Get-as-you-pay
- **Early feedback is generated** because implementation occurs rapidly for a small subset of the software.
- There is an **emphasis on reuse**. This includes the reuse of code, processes, templates, and tools. It is usually faster to assemble pre-built components than to build everything from scratch.
- 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. This avoids the downward flow of the defects.**
- In iterative model **less time is spent on documenting and more time is given for designing**.

Outline

- Iterative Model
- Strengths
- **Weaknesses**
- When to use Iterative Model
- Waterfall vs. Iterative Model

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

Outline

- Iterative Model
- Strengths
- Weaknesses
- **When to use Iterative Model**
- Waterfall vs. Iterative Model

When to use the Iterative Model

- **When risk of long projects cannot be taken**
- **Used commonly in businesses who want quick response for software**
- Most of the **requirements are known up-front but are expected to evolve over time**
- On projects which have **lengthy development schedules**
- **On a project with new technology**
- When the **project is big**.

Outline

- Iterative Model
- Strengths
- Weaknesses
- When to use Iterative Model
- **Waterfall vs. Iterative Model**

Waterfall versus Iterative development Model

- Which software development model(SDLC Model)is best suited for frequently changing requirements ?
- Which software development model(SDLC Model)is best suited when the client's requirements are changing near the release date?

Waterfall versus Iterative development Model

Most people would answer in two words

Iterative Development

Waterfall versus Iterative development Model

- The waterfall has a series of **sequential phases** whereas Iterative model comprises the features of waterfall model in an **iterative manner.**
- An iterative life cycle model **does not attempt to start with a full specification of requirements, whereas waterfall model does.**
- At the end of each waterfall phase, a **"gate review" can be used to determine that the phase has been completed successfully before proceeding with the project. If the phase's deliverables are not acceptable,** the project may simply be stopped or it may return to a prior phase to redefine the project but in iterative we are building and improving the product step by step.
- The common concern about waterfall projects is that **there can be a long delay between project initiation and when the user has their first opportunity to see what's actually been developed**, whereas in iterative projects, product soonly present to the user for feedback.

Outline

- **Evolutionary Process Model**
- Applications
- Advantages
- Disadvantages
- Problems
- Difference between Evolutionary and Iterative Models

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.**
- They are characterized in manner that enables the software engineers to develop increasingly more complete version of a software.
- 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.
- Thus the software engineers can follow a process model that has been explicitly designed to accommodate a product that gradually complete over time.

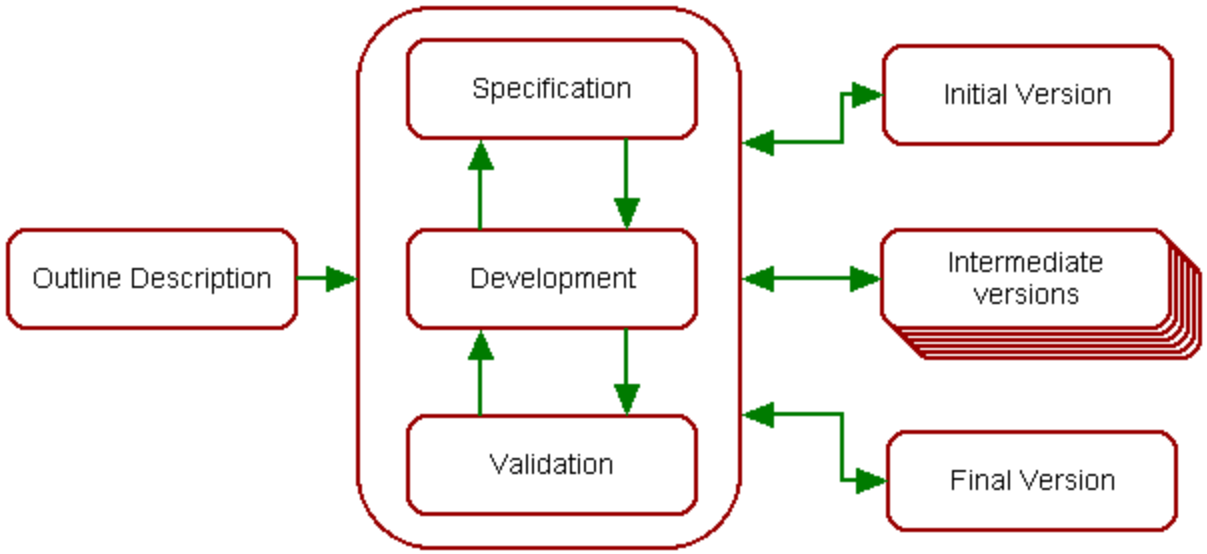
EVOLUTIONARY PROCESS MODEL

Contd…

- **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

Contd…



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

EVOLUTIONARY PROCESS MODEL

Contd…

- 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.

Outline

- Evolutionary Process Model
- **Applications**
- Advantages
- Disadvantages
- Problems
- Difference between Evolutionary and Iterative Models

APPLICATIONS

- For **small or medium-size** interactive systems
- For **parts of large systems** (e.g. the user interface)
- 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**.

Outline

- Evolutionary Process Model
- Applications
- **Advantages**
- Disadvantages
- Problems
- Difference between Evolutionary and Iterative Models

ADVANTAGES

- **Customer involvement** in the process:
  - More likely to meet the user requirement.
- **Deals constantly with changes**
- **Provides quickly an initial version of the system**
- **Involves all development teams**
- **In case major problems are foreseen**, the developer can stop the development after some iterations.
- This model is very **appropriate for research projects**. For example, to develop software for automatic speech recognition, a small vocabulary can be taken and the system is developed. After achieving success, the vocabulary can be increased in stages. This approach is better than starting development of an unlimited vocabulary speech recognition system directly (and after two years, realizing that it is very difficult!).

Outline

- Evolutionary Process Model
- Applications
- Advantages
- **Disadvantages**
- Problems
- Difference between Evolutionary and Iterative Models

DISADVANTAGES

- **Not suitable for large applications**
- Quick fixes may be involved
- **User can get too involved**
- Structure of **system can be damaged since many changes could be made**
- This model **lacks process visibility and offers poor structure** to the project because of the ad-hoc manner of development
- It is **difficult to measure progress and produce documentation reflecting every version of the system as it evolves.** This paradigm usually results in badly structured programs **due to continual code modification**.
- Production of good quality software using this method requires highly skilled and motivated programmers.

Outline

- Evolutionary Process Model
- Types of Evolutionary Development
- Applications
- Advantages
- Disadvantages
- **Problems**
- Difference between Evolutionary and Iterative Models

PROBLEMS

- Lack of process visibility
- Systems are often poorly structured
- Special skills may be required

Outline

- Evolutionary Process Model
- Types of Evolutionary Development
- Applications
- Advantages
- Disadvantages
- Problems
- **Difference between Evolutionary and Iterative Models**

Difference between Evolutionary and Iterative Models

- Evolutionary model differs from the Iterative Enhancement model in the sense that **Evolutionary model doesn't require a usable product at the end of each cycle.** In evolutionary development, requirements are implemented by category rather than by priority.
- The Iterative Enhancement Model is an approach to building software in which the overall lifecycle is compose of several iterations in sequence. The **Evolutionary Enhancement Model is designed to be allowed to evolve in response to the customers feedback.**

Outline

- Evolutionary Process Model
- Types of Evolutionary Development
- Applications
- Advantages
- Disadvantages
- Problems
- Difference between Evolutionary and Iterative Models
- **Types of Evolutionary Models**

Types of Evolutionary Models

- Types of evolutionary models
  - Prototyping Model
  - Spiral Model

Outline

- **Prototyping Model**
- What is Software Prototyping
- Prototyping
- When to use Prototyping Model
- Advantages
- Limitations
- Types of Software Prototyping

PROTOTYPING MODEL

- Prototyping approach, also known as **evolutionary approach**, came to picture because of failures that occurred in the waterfall approach and iterative approach.

- 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.** 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**

Outline

- Prototyping Model
- **What is Software Prototyping**
- Prototyping
- Types of Software Prototyping
- Advantages
- Limitations
- When to use Prototyping Model

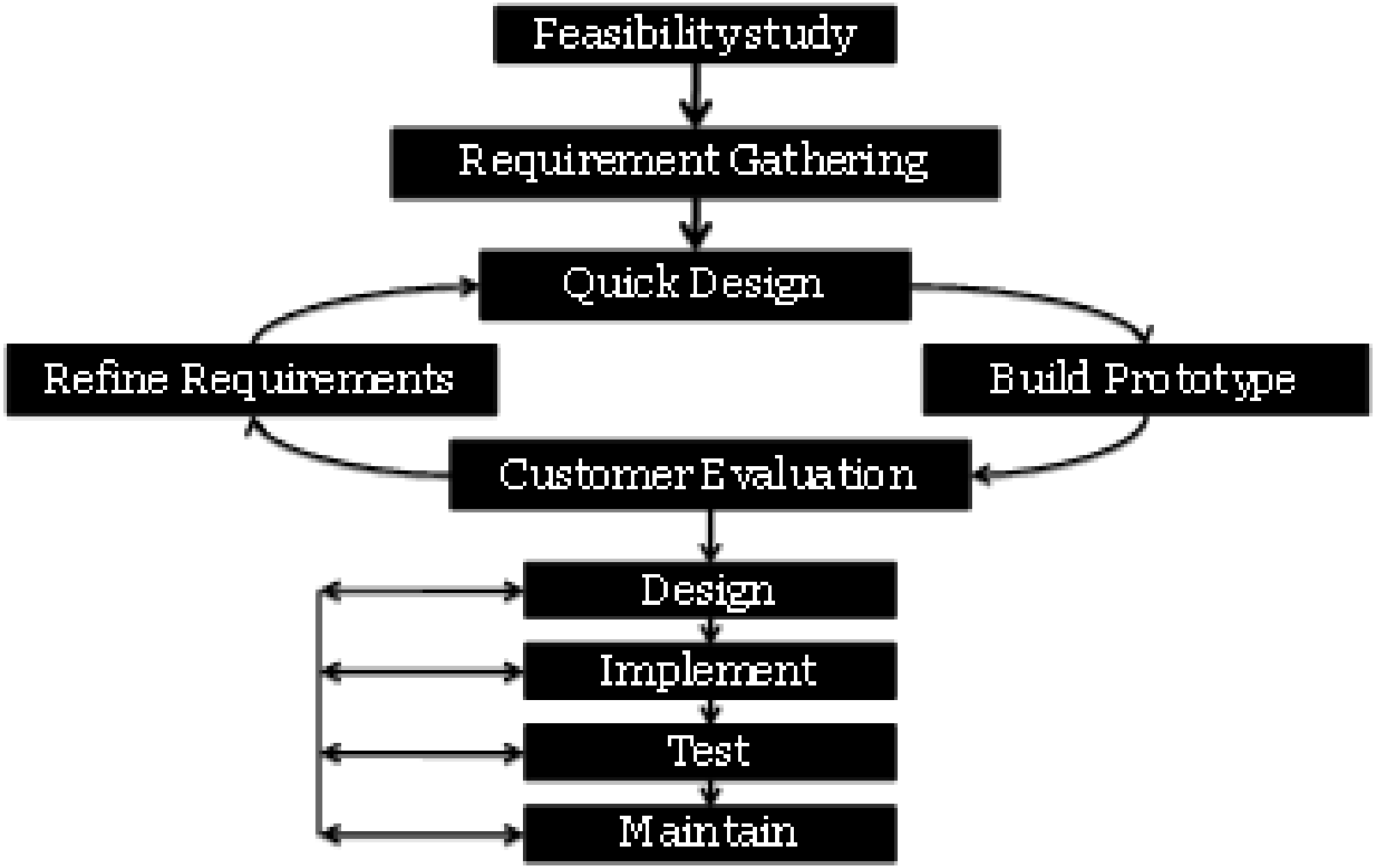
What is Prototype

- 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.**
- A prototype is a **model or a program which is not based on strict planning.**
- **It helps the customer determine how the feature will function in the final software.**
- It is used to **allow the users evaluate developer proposals.**
- It is a **very useful technique to obtain accurate requirements** of the system.

Outline

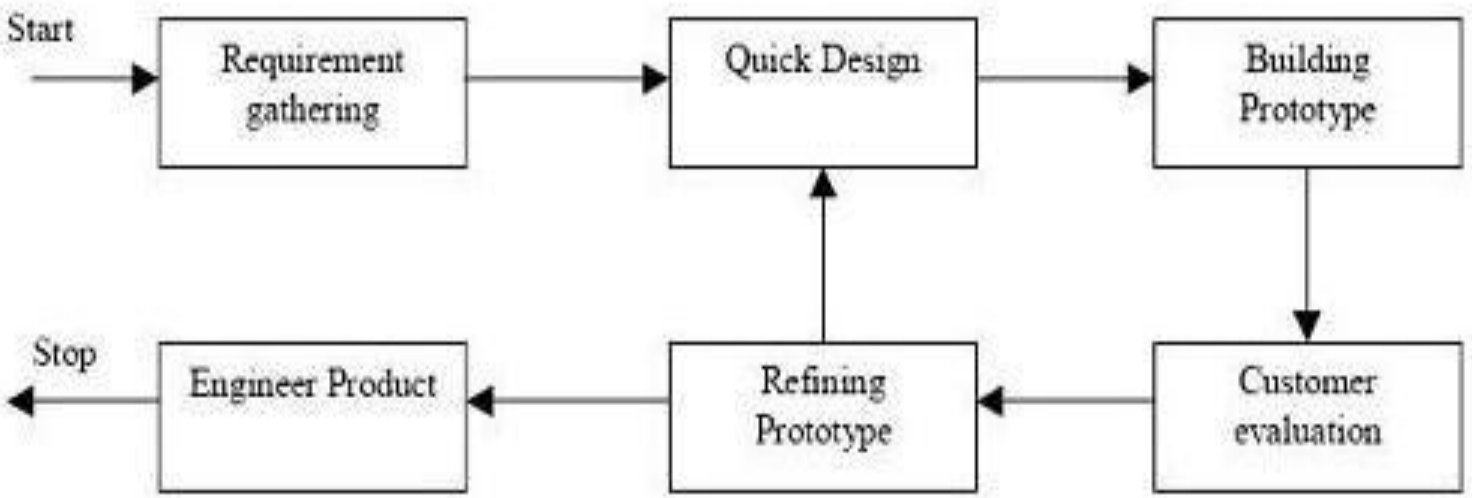
- Prototyping Model
- What is Software Prototyping
- **Prototyping**
- Types of Software Prototyping
- Advantages
- Limitations
- When to use Prototyping Model

Prototyping Model



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. The user is interviewed in order to know the requirements of the system.
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, which gives an idea of the system to the user. A quick design helps in developing the prototype.
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 to recognize its strengths and weaknesses such as what is to be added or removed. 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. That is, a new prototype is developed with the additional information provided by the user. The new prototype is evaluated just like the previous prototype. This process continues until all the requirements specified by the user are met. 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 on regular basis for preventing large-scale failures and minimizing downtime.

Outline

- Prototyping Model
- What is Software Prototyping
- Prototyping
- **Types of Software Prototyping**
- Advantages
- Limitations
- When to use Prototyping Model

TYPES OF SOFTWARE PROTOTYPING

- Throwaway/Rapid Prototyping
- Evolutionary Prototyping
- Incremental Prototyping
- Extreme Prototyping

Outline

- Prototyping Model
- What is Software Prototyping
- Prototyping
- Types of Software Prototyping
- **Advantages**
- Limitations
- When to use Prototyping Model

ADVANTAGES OF PROTOTYPING

- It is a very useful technique to obtain accurate requirements of the system and to **speed up the development process**
- Errors can be detected much earlier.
- Users are actively involved in the development.
- Missing functionality can be identified easily
- 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. They usually are not good at specifying their requirements, nor can tell properly about what they expect from the software.
- **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.
- It reduces risk associated with the software.

Outline

- Prototyping Model
- What is Software Prototyping
- Prototyping
- Types of Software Prototyping
- Advantages
- **Limitations**
- When to use Prototyping Model

LIMITATION OF PROTOTYPING

- In a rush to get it working, overall software quality or long term maintainability are generally overlooked i.e, Use of inappropriate OS or PL or Use of inefficient algorithm
- It gives client a false impression that a few minor changes to the prototype will give them the required system.
- 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.**
- Once we get proper requirements from client after showing prototype model, it may be of no use. That is why, sometimes we refer to the prototype as "**Throw-away" prototype**.
- Too much involvement of client, is not always preferred by the developer.
- The effort invested in building prototypes may be too much if not monitored properly

Outline

- Prototyping Model
- What is Software Prototyping
- Prototyping
- Types of Software Prototyping
- Advantages
- Limitations
- **When to use Prototyping Model**

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**. E.g, 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.

Outline

- **Spiral Model**
- Pictorial Representation of Spiral Model
- Phases of Spiral Model
- Advantages
- Disadvantages
- When to use Spiral Model

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.**
- Include new element : Risk element
- It allows for incremental releases of the product, or incremental refinement through each iteration around the spiral.
- The spiral model, also known as the spiral lifecycle model, is a systems development method (SDM) used in information technology (IT).

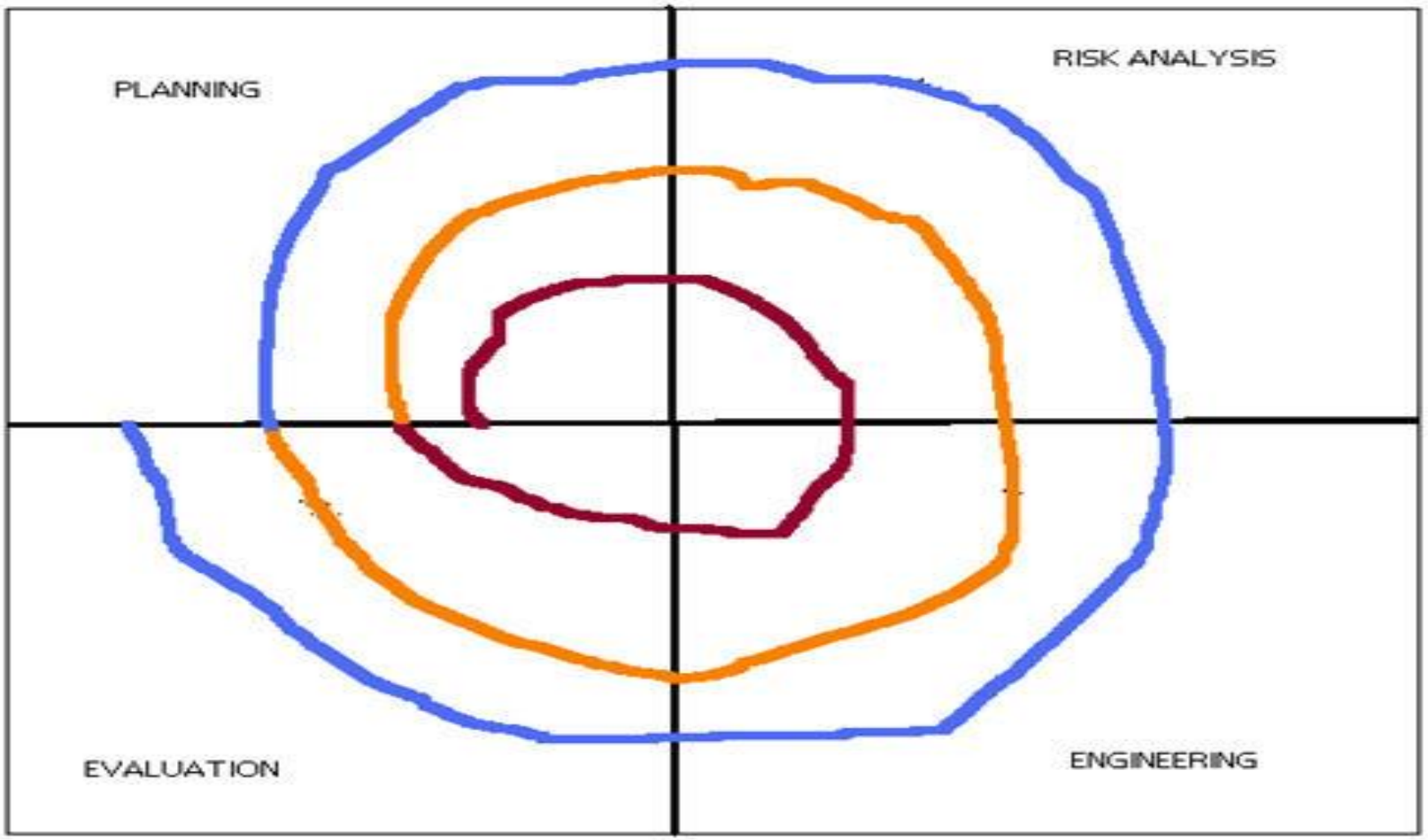
Why Spiral model is called Meta model?

- Because it comprises of other models of SDLC. both waterfall & prototype models are used in it . Here we do software development systematically over the loops & at the same time we make a prototype & show it to user after completion of various phase. This way we are able to reduce risks as well as follow systematic approach.

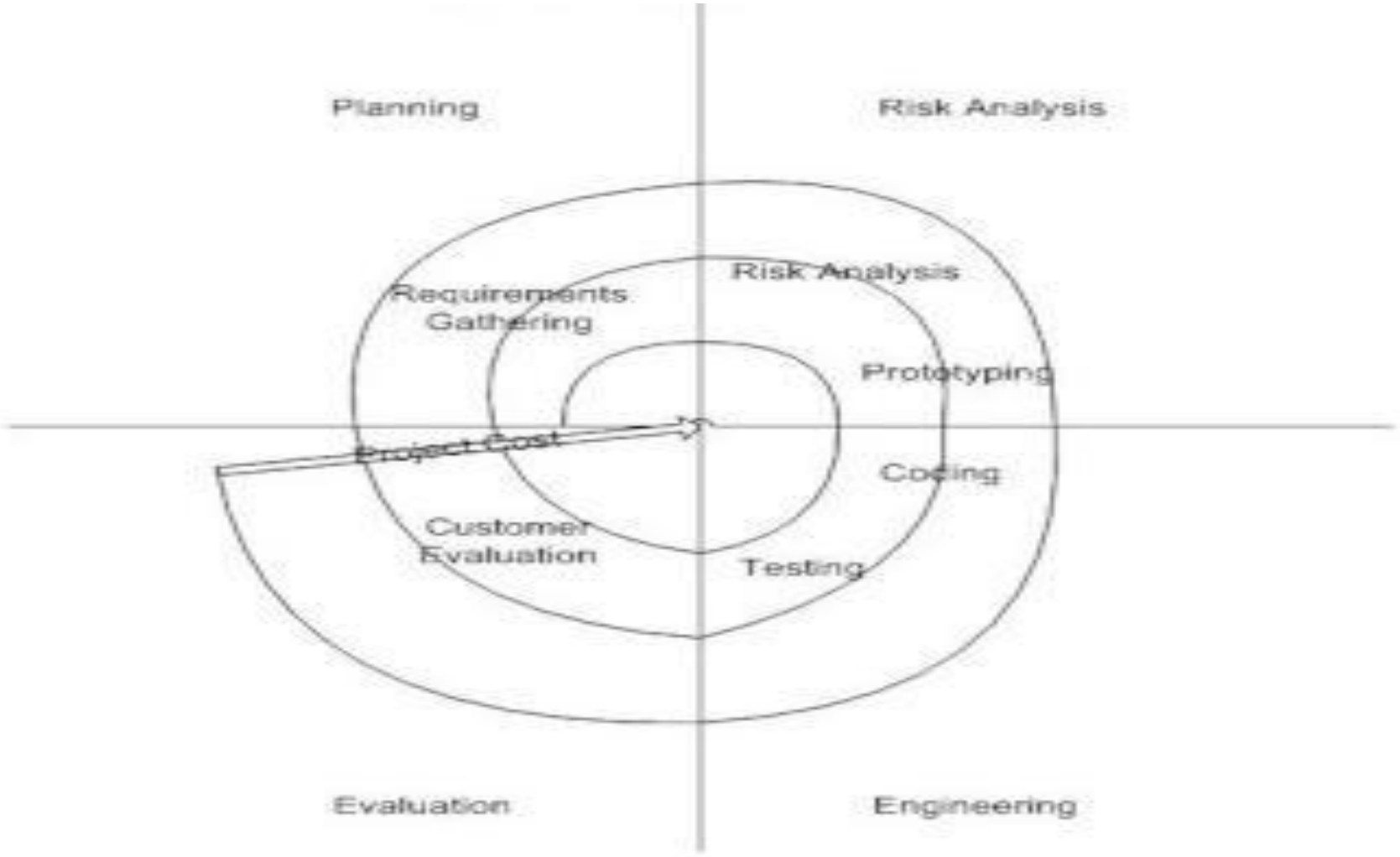
THE SPIRAL MODEL

- 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



SDLC Spiral Model



Phases of Spiral Model

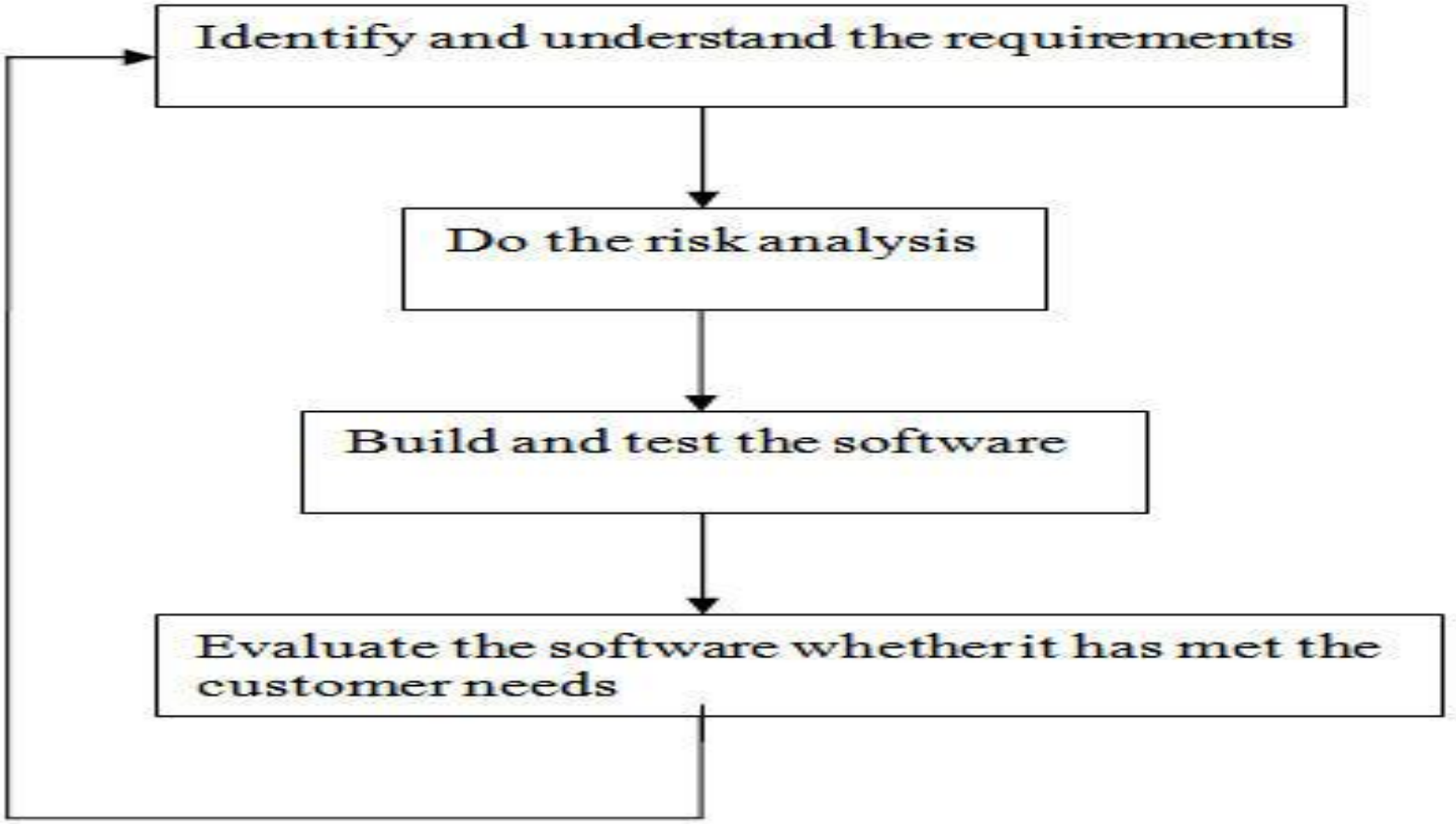
- **Planning Phase:** Requirements are gathered during the planning phase. Requirements like 'BRS' that is 'Bussiness Requirement Specifications' and 'SRS' that is 'System Requirement specifications'.

- **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.

In simpler terms, steps involved in Spiral Model



SDLC Spiral Model - © www.SoftwareTestingHelp.com

Advantages

- High amount of risk analysis hence, avoidance of Risk is enhanced.
- **The spiral model is favored for large, expensive, and complicated projects.**
- Additional Functionality can be added at a later date.
- Development can be divided into smaller parts and more risky parts can be developed earlier which helps better risk management.
- Development is fast
- Has room for customer feedback and the changes are implemented faster.
- Better understanding between developer and customer

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.
- Management is more complex.
- Not suitable for small or low risk projects and could be expensive for small projects.
- Spiral may go indefinitely(infinitely).

When to use Spiral Model

- When there is a budget constraint and risk evaluation is important.
- Requirements are complex and need evaluation to get clarity.
- Significant changes are expected(research and exploration) in the product during the development cycle.
- When the project is large.
- Where enough time frame is their to get end user feedback.
- Where releases are required to be frequent.

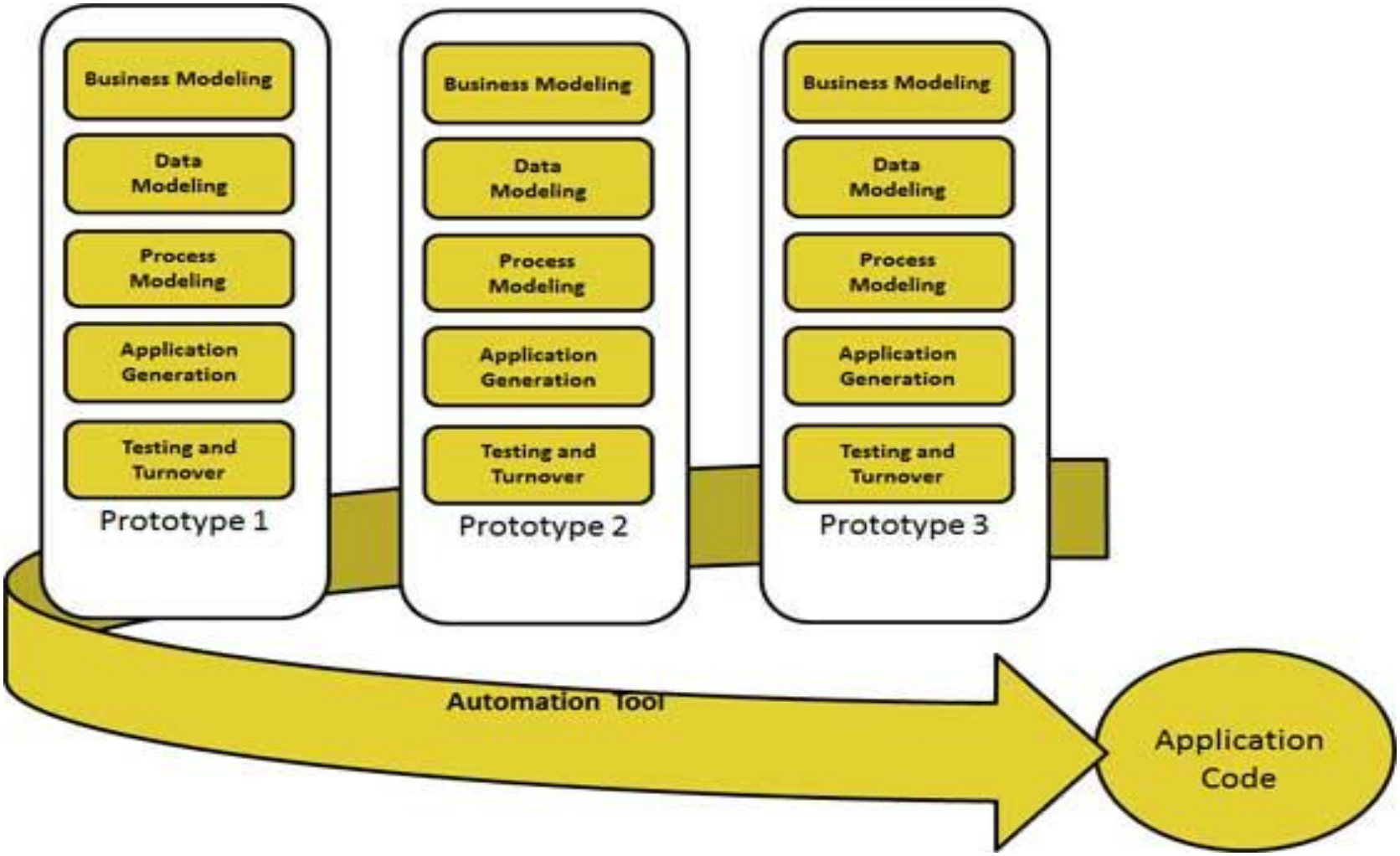
Outline

- RAD Model
- Applications
- Pros
- Cons

WHAT IS RAD

- RAD model is Rapid Application Development model.
- It is a **based on prototyping and iterative development.**
- In RAD the Components are developed in parallel Manner.
- It is a faster software development process.

RAD MODEL


Business Modeling
Data Modeling
Process Modeling
Application Generation
Testing and Turnover
Prototype 1
Business Modeling
Data Modeling
Process Modeling
Application Generation
Testing and Turnover
Prototype 2
Business Modeling
Data Modeling
Process Modeling
Application Generation
Testing and Turnover
Prototype 3
Automation Tool
Application Code

Business Modeling

The business model for the product under development is designed in terms of flow of information and the distribution of information between various business channels. A complete business analysis is performed to find the vital information for business, how it can be obtained, how and when is the information processed and what are the factors driving successful flow of information.

Data Modeling

The information gathered in the Business Modeling phase is reviewed and analyzed to form sets of data objects vital for the business. The attributes of all data sets is identified and defined. The relation between these data objects are established and defined in detail in relevance to the business model.

Process Modeling

The data object sets defined in the Data Modeling phase are converted to establish the business information flow needed to achieve specific business objectives as per the business model. The process model for any changes or enhancements to the data object sets is defined in this phase. Process descriptions for adding , deleting, retrieving or modifying a data object are given.

Application Generation

The actual system is built and coding is done by using automation tools to convert process and data models into actual prototypes.

Testing and Turnover

The overall testing time is reduced in RAD model as the prototypes are independently tested during every iteration. However the data flow and the interfaces between all the components need to be thoroughly tested with complete test coverage. Since most of the programming components have already been tested, it reduces the risk of any major issues.

Applications

- RAD should be used only when a system can be modularized to be delivered in incremental manner.
- It should be used only if the budget permits use of automated code generating tools.
- RAD SDLC model should be chosen only if domain experts are available with relevant business knowledge.
- Should be used where the requirements change during the course of the project and working prototypes are to be presented to customer in small iterations of 2-3 months.

Pros

- RAD model enables rapid delivery as it reduces the overall development time due to reusability of the components and parallel development.
- RAD works well only if high skilled engineers are available and the customer is also committed to achieve the targeted prototype in the given time frame. If there is commitment lacking on either side the model may fail.
- Changing requirements can be accommodated.
- Iteration time can be short with use of powerful RAD tools.
- Reduced development time.
- Increases reusability of components
- Quick initial reviews occur

Cons

- Dependency on technically strong team members for identifying business requirements.
- Only system that can be modularized can be built using RAD.
- Requires highly skilled developers/designers.
- Inapplicable to cheaper projects as cost of modeling and automated code generation is very high.
- Suitable for systems that are component based and scalable.

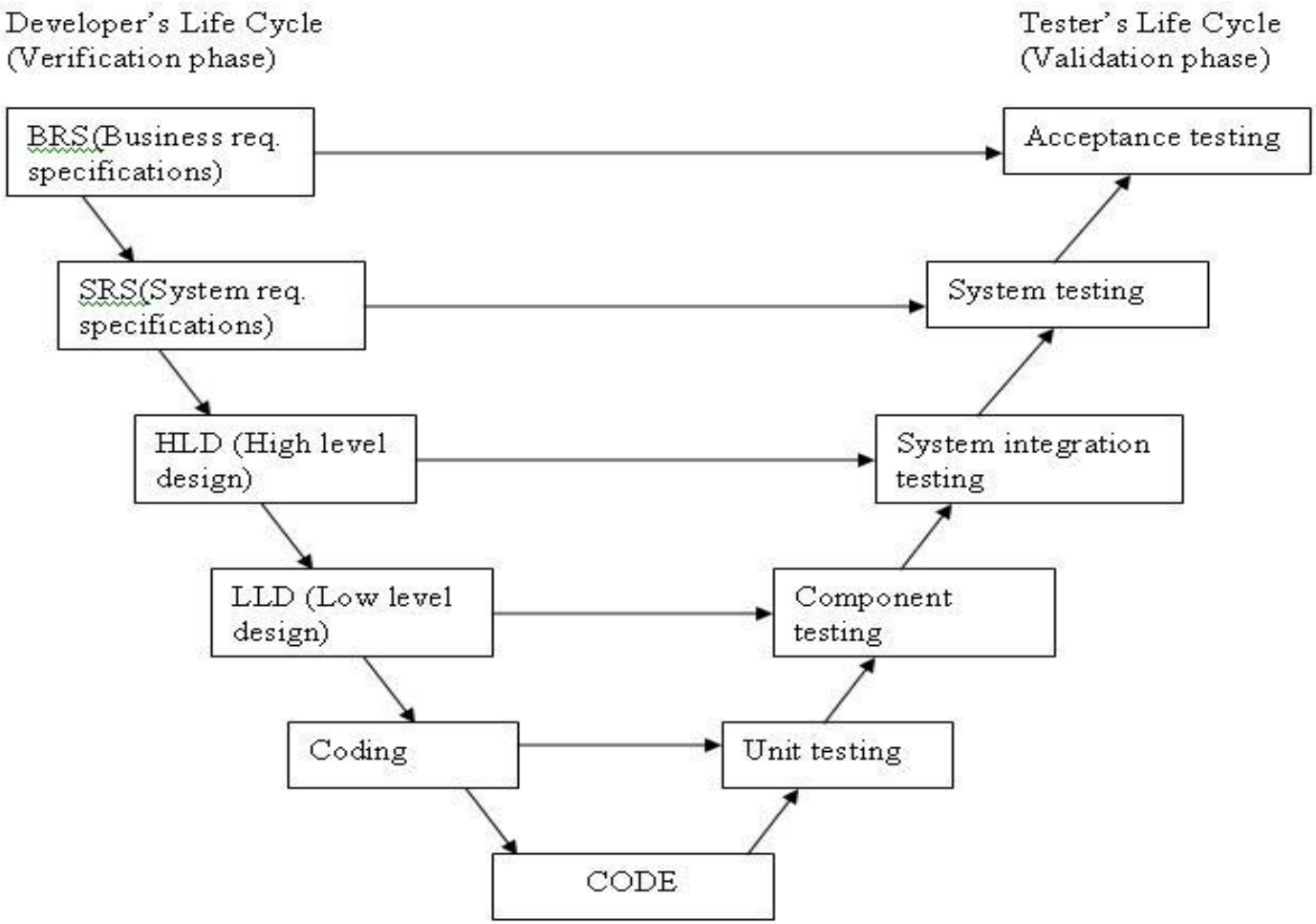
Outline

- V Model
- Advantages
- Disadvantages
- When to use V Model

V MODEL

- V- model means Verification and Validation model.
- Just like the waterfall model, the V-Shaped life cycle is a sequential path of execution of processes.
- Each phase must be completed before the next phase begins.
- Testing of the product is planned in parallel with a corresponding phase of development.

Diagram of V Model



ADVANTAGES

- Simple and easy to use.
- Testing activities like planning, test designing happens well before coding. This saves a lot of time. Hence higher chance of success over the waterfall model.
- Proactive defect tracking – that is defects are found at early stage.
- Avoids the downward flow of the defects.
- Works well for small projects where requirements are easily understood.

DISADVANTAGES

- Requirements are well defined, clearly documented and fixed.
- Product definition is stable.
- Technology is not dynamic and is well understood by the project team.
- There are no ambiguous or undefined requirements.
- The project is short.

When to use the V-Shaped Model

- All requirements are known up-front
- Solution and technology are known
- Project is short.
- High confidence of customer is required for choosing the V-shaped model approach. Since, no prototypes are produced, there is very high risk involved in meeting customer expectations.

END OF UNIT-1