**Software Engineering Concepts: Assignment 1**

**Aim:**

To study Software Development Life Cycle ( SDLC) Models

**Team Number and Team Name:**

**Team No 7. Code Panther**

**Team Members:**

1. Pranali Wandhare

2. Pranav Mirajgaonkar

3.Pranav Salwaru

4. Prasad Kunte

5. Prashant Tambe

6. Prashant Dhole

7. Pratiksha More

8. Priya Janghele

9. Pushkar Attarde

10. Rahul Deore

**Lab Co-ordinator:**

Nisha Karoli

**Software Development Life Cycle ( SDLC) Models:**

1. **Waterfall Model**

The classical waterfall model is the basic **software development life cycle** model. It is very simple but idealistic. Earlier this model was very popular but nowadays it is not used. But it is very important because all the other software development life cycle models are based on the classical waterfall model.   
The classical waterfall model divides the life cycle into a set of phases. This model considers that one phase can be started after the completion of the previous phase. That is the output of one phase will be the input to the next phase. Thus the development process can be considered as a sequential flow in the waterfall. Here the phases do not overlap with each other. The different sequential phases of the classical waterfall model are shown in the below figure:



1. **Feasibility Study**: The main goal of this phase is to determine whether it would be financially and technically feasible to develop the software.   
   The feasibility study involves understanding the problem and then determining the various possible strategies to solve the problem. These different identified solutions are analyzed based on their benefits and drawbacks, The best solution is chosen and all the other phases are carried out as per this solution strategy.
2. **Requirements analysis and specification**: The aim of the requirement analysis and specification phase is to understand the exact requirements of the customer and document them properly. This phase consists of two different activities.
   * **Requirement gathering and analysis:** Firstly all the requirements regarding the software are gathered from the customer and then the gathered requirements are analyzed. The goal of the analysis part is to remove incompleteness (an incomplete requirement is one in which some parts of the actual requirements have been omitted) and inconsistencies (an inconsistent requirement is one in which some part of the requirement contradicts some other part).
   * **Requirement specification:** These analyzed requirements are documented in a software requirement specification (SRS) document. SRS document serves as a contract between the development team and customers. Any future dispute between the customers and the developers can be settled by examining the SRS document.
3. **Design**: The goal of this phase is to convert the requirements acquired in the SRS into a format that can be coded in a programming language. It includes high-level and detailed design as well as the overall software architecture. A Software Design Document is used to document all of this effort (SDD)
4. **Coding and Unit testing**: In the coding phase software design is translated into source code using any suitable programming language. Thus each designed module is coded. The aim of the unit testing phase is to check whether each module is working properly or not.
5. **Integration and System testing**: Integration of different modules are undertaken soon after they have been coded and unit tested. Integration of various modules is carried out incrementally over a number of steps. During each integration step, previously planned modules are added to the partially integrated system and the resultant system is tested. Finally, after all the modules have been successfully integrated and tested, the full working system is obtained and system testing is carried out on this.

**Purpose:**

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

• 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.

**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.

**Real Time Example:**

• Use to develop enterprise applications like Customer Relationship Management (CRM) systems

• Human Resource Management Systems (HRMS)

• Supply Chain Management Systems

• Inventory Management Systems

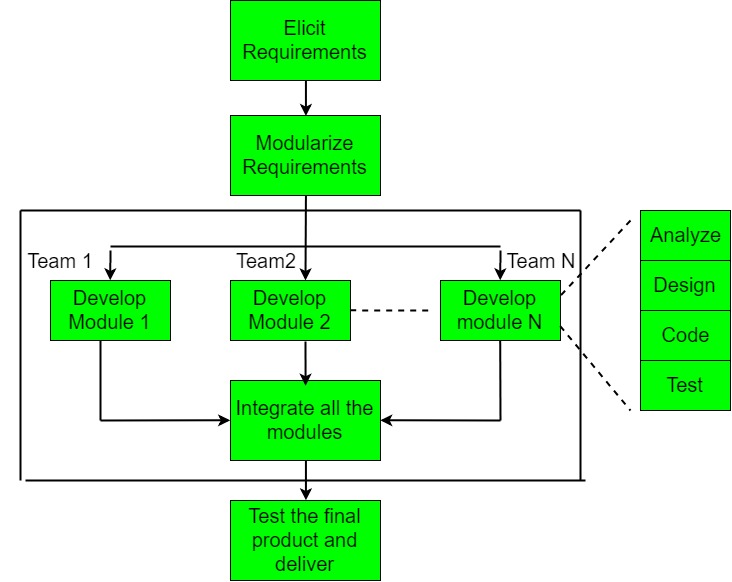
• Point of Sales (POS) systems for Retail chains

• Development of Department Of Defence (DOD)

• military and aircraft programs followed Waterfall model in many organizations

1. **Rapid Application Development**

The Rapid Application Development Model was first proposed by IBM in the 1980s. The critical feature of this model is the use of powerful development tools and techniques. A software project can be implemented using this model if the project can be broken down into small modules wherein each module can be assigned independently to separate teams. These modules can finally be combined to form the final product. Development of each module involves the various basic steps as in the waterfall model i.e analyzing, designing, coding, and then testing, etc. as shown in the figure. Another striking feature of this model is a short time span i.e the time frame for delivery(time-box) is generally 60-90 days.



The use of powerful developer tools such as JAVA, C++, Visual BASIC, XML, etc. is also an integral part of the projects. This model consists of 4 basic phases:

1. **Requirements Planning –** It involves the use of various techniques used in requirements elicitation like brainstorming, task analysis, form analysis, user scenarios, FAST (Facilitated Application Development Technique), etc. It also consists of the entire structured plan describing the critical data, methods to obtain it, and then processing it to form a final refined model.
2. **User Description –** This phase consists of taking user feedback and building the prototype using developer tools. In other words, it includes re-examination and validation of the data collected in the first phase. The dataset attributes are also identified and elucidated in this phase.
3. **Construction –** In this phase, refinement of the prototype and delivery takes place. It includes the actual use of powerful automated tools to transform process and data models into the final working product. All the required modifications and enhancements are too done in this phase.
4. **Cutover –** All the interfaces between the independent modules developed by separate teams have to be tested properly. The use of powerfully automated tools and subparts makes testing easier. This is followed by acceptance testing by the user.

The process involves building a rapid prototype, delivering it to the customer, and taking feedback. After validation by the customer, the SRS document is developed and the design is finalized.

**Advantages:**

* The use of reusable components helps to reduce the cycle time of the project.
* Feedback from the customer is available at the initial stages.
* Reduced costs as fewer developers are required.
* The use of powerful development tools results in better quality products in comparatively shorter time spans.
* The progress and development of the project can be measured through the various stages.
* It is easier to accommodate changing requirements due to the short iteration time spans.
* The use of powerful and efficient tools requires highly skilled professionals.
* The absence of reusable components can lead to the failure of the project

**Disadvantages:**

The team leader must work closely with the developers and customers to close the project in time.

* The systems which cannot be modularized suitably cannot use this model.
* Customer involvement is required throughout the life cycle.
* It is not meant for small-scale projects as in such cases, the cost of using automated tools and techniques may exceed the entire budget of the project

**Purpose:**

• RAD should be used when there is a need to create a system that can be modularized in 2-3 months of time.

• It should be used if there’s high availability of designers for modeling and the budget is high enough to afford their cost along with the cost of automated code generating tools.

• RAD SDLC model should be chosen only if resources with high business knowledge are available and there is a need to produce the system in a short span of time (2-3 months).

**Real Time Example:**

• General Goals - Bonus Subsystem

• Current System - Vehicle Subsystem

• Proposed System - VIP Subsystem

➢ User Interface and Human Factors - Maintenance Subsystem

➢ Documentation - Travel Subsystem

➢ Hardware Consideration - Logbook Subsystem

➢ Performance Characteristics - Bonus Subsystem

➢ Screen Mock up - Maintenance Subsystem

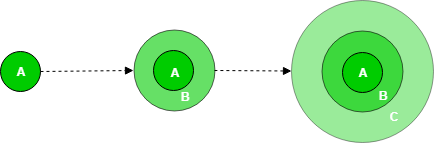
➢ Navigational Path for the Web Application - Bonus Subsystem

➢ Dynamic Models - Logbook Subsystem

**3.3 Incremental Model**

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.

**Life cycle activities:**  
Requirements of Software are first broken down into several modules that can be incrementally constructed and delivered. At any time, the plan is made just for the next increment and not for any kind of long-term plan. Therefore, it is easier to modify the version as per the need of the customer. The Development Team first undertakes to develop core features (these do not need services from other features) of the system.

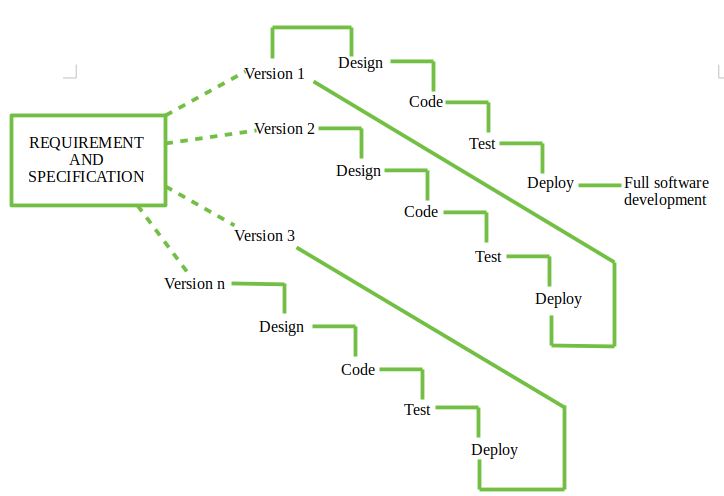
Once the core features are fully developed, then these are refined to increase levels of capabilities by adding new functions in Successive versions. Each incremental version is usually developed using an iterative waterfall model of development.

As each successive version of the software is constructed and delivered, now the feedback of the Customer is to be taken and these were then incorporated into the next version. Each version of the software has more additional features than the previous ones.

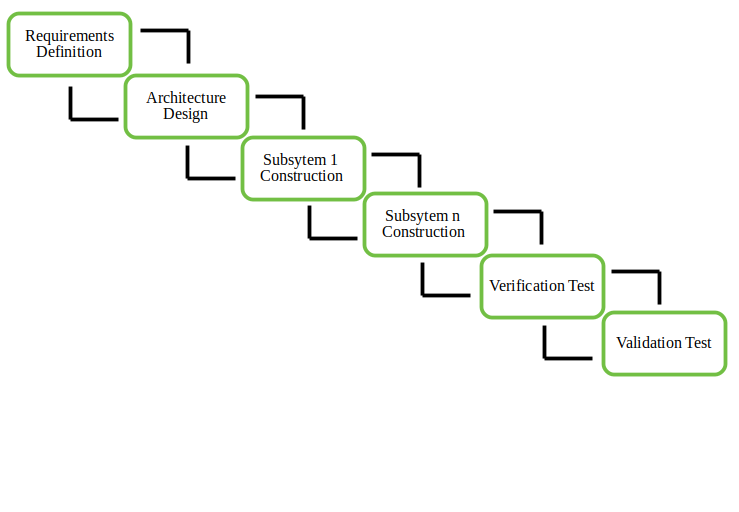
**Disadvantages:**

After Requirements gathering and specification, requirements are then split into several different versions starting with version 1, in each successive increment, the next version is constructed and then deployed at the customer site. After the last version (version n), it is now deployed at the client site.

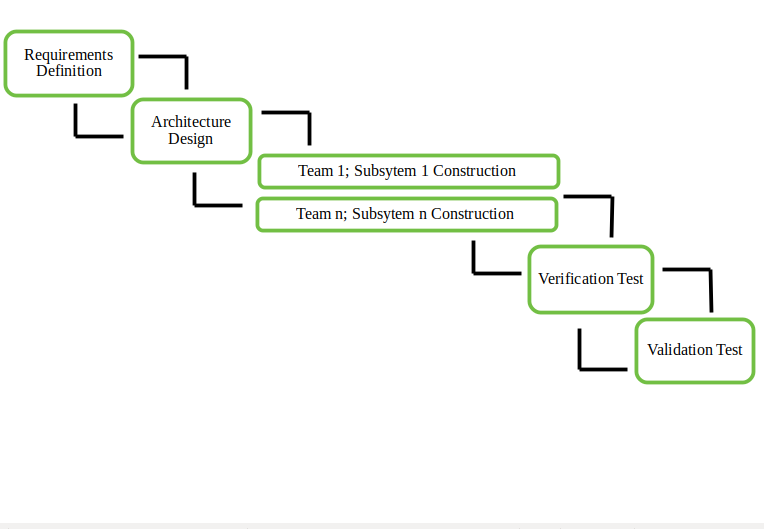
**Types of Incremental model:**



**1. Staged Delivery Model:**Construction of only one part of the project at a time.



**2. Parallel Development Model –**Different subsystems are developed at the same time. It can decrease the calendar time needed for the development, i.e. TTM (Time to Market) if enough resources are available.



**When to use this:**

1. Funding Schedule, Risk, Program Complexity, or need for early realization of benefits.

2. When Requirements are known up-front.

3. When Projects have lengthy development schedules.

4. Projects with new Technology.

* Error Reduction (core modules are used by the customer from the beginning of the phase and then these are tested thoroughly)
* Uses divide and conquer for a breakdown of tasks.
* Lowers initial delivery cost.
* Incremental Resource Deployment.

5. Requires good planning and design.

6. The total cost is not lower.

7. Well-defined module interfaces are required.

**Advantages-**

1. Prepares the software fast.
2. Clients have a clear idea of the project.
3. Changes are easy to implement.
4. Provides risk handling support, because of its iterations.

**Disadvantages-**

1. A good team and proper planned execution are required.
2. Because of its continuous iterations the cost increases.

**Purpose:**

• This model can be used when the requirements of the complete system are clearly defined and understood.

• Major requirements must be defined; however, some details can evolve with time.

• There is a need to get a product to the market early.

• A new technology is being used

• Resources with needed skill set are not available

• There are some high risk features and goals.

• This model is more flexible – less costly to change scope and requirement. **Real Time Example:**

• This model can be used when the requirements of the complete system are clearly defined and understood.

• Major requirements must be defined; however, some details can evolve with time.

• There is a need to get a product to the market early.

• A new technology is being used.

• Resources with needed skill set are not available.

• There are some high-risk features and goals.

**4. Spiral Model**

**Spiral model** is one of the most important Software Development Life Cycle models, which provides support for **Risk Handling**. In its diagrammatic representation, it looks like a spiral with many loops. The exact number of loops of the spiral is unknown and can vary from project to project. Each loop of the spiral is called a **Phase of the software development process.** The exact number of phases needed to develop the product can be varied by the project manager depending upon the project risks. As the project manager dynamically determines the number of phases, so the project manager has an important role to develop a product using the The Radius of the spiral at any point represents the expenses(cost) of the project so far, and the angular dimension represents the progress made so far in the current phase.

**The below diagram shows the different phases of the Spiral Model: –**



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

**Purpose:**

* • 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

**Advantages of Spiral Model**:   
Below are some advantages of the Spiral Model.

1. **Risk Handling:** The projects with many unknown risks that occur as the development proceeds, in that case, Spiral Model is the best development model to follow due to the risk analysis and risk handling at every phase.
2. **Good for large projects:** It is recommended to use the Spiral Model in large and complex projects.
3. **Flexibility in Requirements:** Change requests in the Requirements at later phase can be incorporated accurately by using this model.
4. **Customer Satisfaction:** Customer can see the development of the product at the early phase of the software development and thus, they habituated with the system by using it before completion of the total product.

**Disadvantages of Spiral Model**:   
Below are some main disadvantages of the spiral model.

1. **Complex:** The Spiral Model is much more complex than other SDLC models.
2. **Expensive:** Spiral Model is not suitable for small projects as it is expensive.
3. **Too much dependability on Risk Analysis:** The successful completion of the project is very much dependent on Risk Analysis. Without very highly experienced experts, it is going to be a failure to develop a project using this model.
4. **Difficulty in time management:** As the number of phases is unknown at the start of the project, so time estimation is very difficult.

**Real Time Example:**

• Working on the missiles or satellites is the real time example of a spiral model.

• The spiral model uses the approach of Prototyping Model by building a prototype at the start of each phase as a risk handling technique.

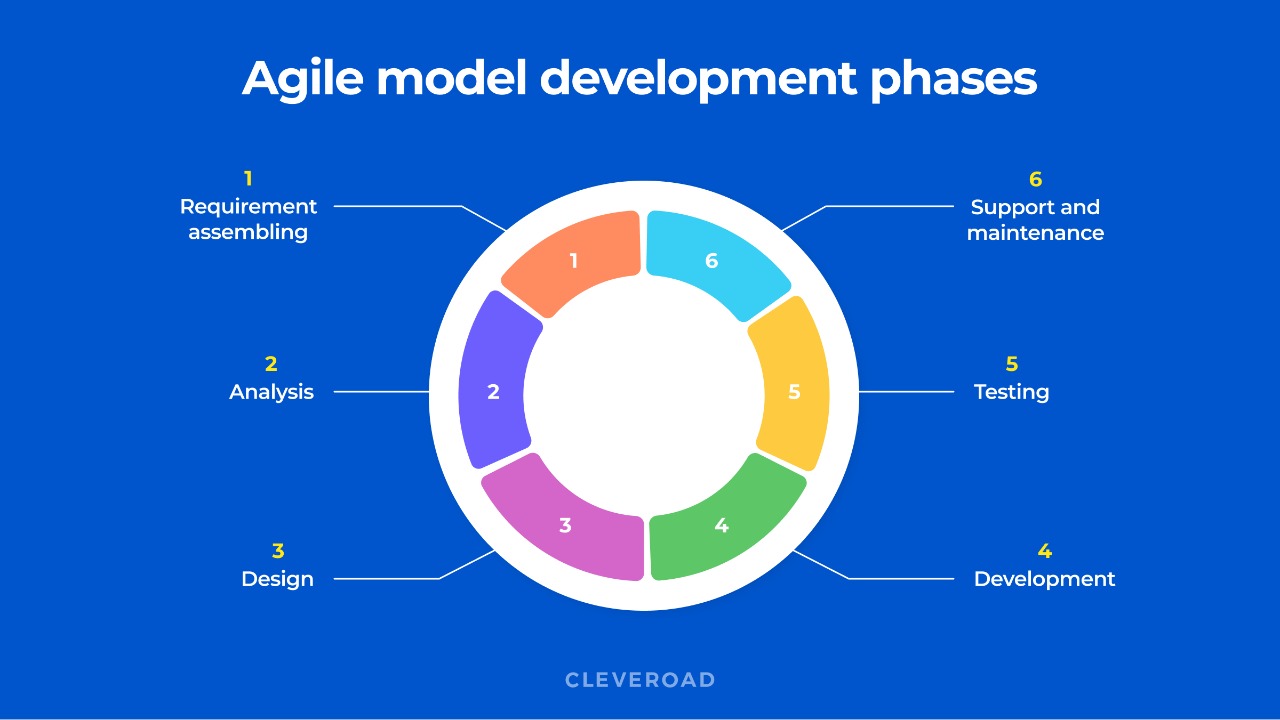
• Gantt chart software – GanttPRO a tool for simple task handling.

• Evolution of Microsoft Windows operating system.

**5. Agile Model**

The Agile model was primarily designed to help a project to adapt to change requests quickly. So, the main aim of the Agile model is to facilitate quick project completion. To accomplish this task agility is required. Agility is achieved by fitting the process to the project, removing activities that may not be essential for a specific project. Also, anything that is waste of time and effort is avoided.

Actually Agile model refers to a group of development processes. These processes share some basic characteristics but do have certain subtle differences among themselves. A few Agile SDLC models are given below:



Agile model is the combination of iterative and incremental process models. The steps involve in agile SDLC models are:

* Requirement gathering
* Requirement Analysis
* Design
* Coding
* Unit testing
* Acceptance testing

The time to complete an iteration is known as a Time Box. Time-box refers to the maximum amount of time needed to deliver an iteration to customers. So, the end date for an iteration does not change. Though the development team can decide to reduce the delivered functionality during a Time-box if necessary to deliver it on time. The central principle of the Agile model is the delivery of an increment to the customer after each Time-box.

**Advantages:**

* Working through Pair programming produce well written compact programs which have fewer errors as compared to programmers working alone.
* It reduces total development time of the whole project.
* Customer representatives get the idea of updated software products after each iteration. So, it is easy for him to change any requirement if needed.

**Disadvantages:**

* Due to lack of formal documents, it creates confusion and important decisions taken during different phases can be misinterpreted at any time by different team members.
* Due to the absence of proper documentation, when the project completes and the developers are assigned to another project, maintenance of the developed project can become a problem.
* **Real Time Example:**
* • Restaurant orders:
* ➢ Preparation of some of the food before opening the shop (sprint planning)
* ➢ continuous delivery of orders (adhoc stories)
* ➢ number of successful orders (velocity)
* • Cricket:
* ➢ over (sprint length)
* ➢ team (scrum team self-sufficient)
* ➢ Run rate (velocity)
* ➢ Captain/ coach (scrum master)
* **6. ‘V’ Model:**
* The V-model is a type of SDLC model where process executes in a sequential manner in V-shape. It is also known as Verification and Validation model. It is based on the association of a testing phase for each corresponding development stage. Development of each step directly associated with the testing phase. The next phase starts only after completion of the previous phase i.e. for each development activity, there is a testing activity corresponding to it.



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

**Testing Phases:** 

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

**Advantages:** 

* This is a highly disciplined model and Phases are completed one at a time.
* V-Model is used for small projects where project requirements are clear.
* Simple and easy to understand and use.
* This model focuses on verification and validation activities early in the life cycle thereby enhancing the probability of building an error-free and good quality product.
* It enables project management to track progress accurately.

**Disadvantages:** 

* High risk and uncertainty.
* It is not a good for complex and object-oriented projects.
* It is not suitable for projects where requirements are not clear and contains high risk of changing.
* This model does not support iteration of phases.
* It does not easily handle concurrent events.

**Purpose:**

• The V-shaped model should be used for small to medium sized projects where requirements are clearly defined and fixed.

• The V-Shaped model should be chosen when ample technical resources are available with needed technical expertise.

**Real Time Example:**

• IT projects by federal agencies

• public-sector software projects

• In electronic and mechanical system in research and science

• software for agencies and ministries

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |