**SDLC MODELS**

1. ***WATERFALL MODEL***

**ABOUT WATER FALL MODEL:**

* It is a traditional model.
* It is a sequential design process that is often used in SDLC(Software Development Life Cycle)
* Progress of this model is seen as flowing downward like a waterfall through different phases.
* Such as,
* This methodology is preferred in project where quality is more important as compared to schedule or cost.
* This methodology is suitable for short term project, where the requirements will not change.
* Ex- Your attendance management Calculator.

**BENIFITS OF WATER FALL MODEL**

1. Very simple to implement**.**

2. Requirement do not change in order to design and cost.

3. We can get stable products.

4. Requirements will finalize earlier in the SDLC there won’t be any confusion in the next phase.

5. To implement this model is Minimal compared to other methodologies.

6. Every phases has specific deliverables it gives high visibility to project manager & clients about the progress of the project.

**DISADVANTAGES OF WATER FALL MODEL**

1. The main disadvantage of this model is **Back tracking is not possible.**

2. We cannot go back & change the requirements once the design stage is reached**.**

***2. 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.

**When to use the Iterative Model?**

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

**Advantages of Iterative Model:**

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

**Disadvantage of Iterative Model:**

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

***3. SPIRAL MODEL***

* **It is the combination of waterfall model & iterative model.**
* **This model adapts large and complicated projects where risk is very high.**
* **Every iteration in spiral model begin with design goal & ends with product evaluation by client.**
* **Ex:-product development team like Microsoft.**
* **They know that there will be a high risk & they face lot of difficulties in the journey of developing & releasing a project.**

**Also they know that, will release next version of product when the current version is in existence.**

**They prefer spiral model so develop the product in an iterative nature.**

**They could releasing one version of a product to the end user & start developing next version which includes new enhancement & improvement on previous version.**

**That is based on issues faced by the user.**

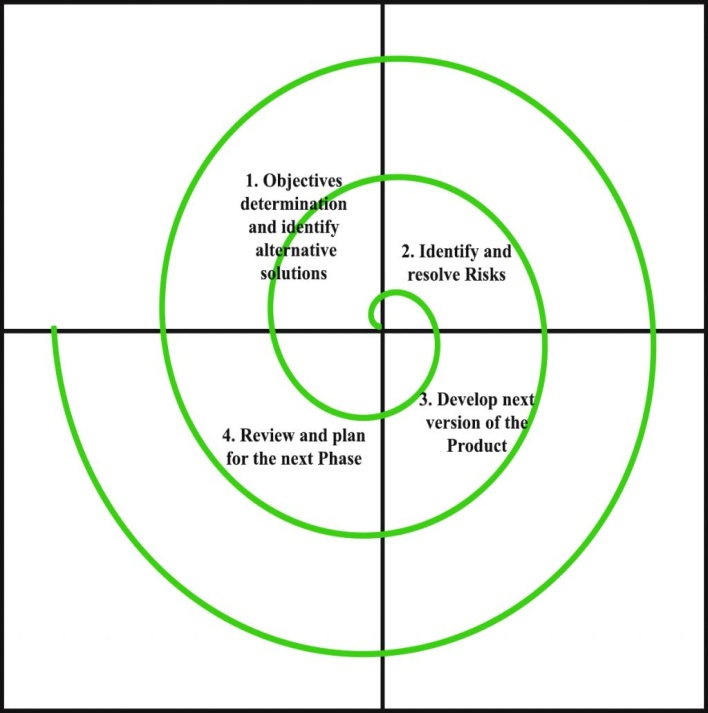
**Spiral model undergoes 4 phases,**

**1. Planning phase**

**2. Risk Analysis phase**

**3. Design or engineering phase**

**4. Evaluation phase**

****

**Planning phase:** Requirement gathering, cost estimation & resource allocation

**Risk analysis phase:** strengths & weakness of the projects are being known

**Design & development phase:** Their coding internal testing & deployment is being done.

**Evaluation phase:** client evaluation to get the feedback

**Benefits of Spiral model:-**

1. Allows requirements changes

2. suitable for large & complicated project

3. It allows better risk analysis & cost effective due to good risk management.

**Disadvantages of spiral model:-**

1. Not suitable for small projects
2. Success of a project depends on Risk analysis phase

***4. V MODEL***

* The V-model is an SDLC model where execution of processes happens in a sequential manner in a V-shape.
* It is also known as **Verification and Validation model**.
* The V-Model is an extension of the waterfall model.
* Under the V-Model, the corresponding testing phase of the development phase is planned in parallel.
* So, there are Verification phases on one side of the ‘V’ and Validation phases on the other side.
* The Coding Phase joins the two sides of the V-Model.

**V MODEL:**

## SDLC V-Model

## Verification Phases

### Business Requirement Analysis

1. Product requirements are understood from the customer’s perspective.
2. This phase involves detailed communication with the customer to understand his expectations and exact requirement.
3. This is a very important activity and needs to be managed well, as most of the customers are not sure what exactly they need.
4. The **acceptance test design planning** is done at this stage as business requirements can be used as an input for acceptance testing.

### System Design

1. The system design will have the understanding and detailing the complete hardware and communication setup for the product under development.
2. The system **test plan is developed** based on the system design.

### Architectural Design

1. The system design is broken down further into ***modules*** taking up different functionality.
2. This is also referred to as **High Level Design (HLD)**.
3. The data transfer and communication between the internal modules and with the outside world (other systems) is clearly understood and defined in this stage.
4. With this information, **integration tests can be designed** and documented during this stage.

### Module Design

1. In this phase, the detailed internal design for all the system modules is specified, referred to as **Low Level Design (LLD)**.

2. The unit tests are an essential part of any development process and **helps eliminate the maximum faults and errors at a very early stage.**

3. These unit tests can be designed at this stage based on the internal module designs.

## Coding Phase

1. The actual coding of the system modules designed in the design phase is taken up in the Coding phase.

2. The **best suitable programming language is decided** based on the system and architectural requirements.

3. The coding is performed based on the coding guidelines and standards.

## Validation Phases

## Unit Testing

## 1. Unit tests designed in the module design phase are executed on the code during this validation phase.

## 2. Unit testing is the testing at code level and helps eliminate bugs at an early stage

### Integration Testing

1. Integration testing is associated with the **architectural design phase.**

2. Integration tests are performed to test the co-existence and communication of the internal modules within the system.

### System Testing

1. System testing is directly associated with the **system design phase.**

2. System tests check the entire system functionality and the communication of the system under development with external systems.

3. Most of the software and hardware compatibility issues can be uncovered during this system test execution.

### Acceptance Testing

1. Acceptance testing is associated with the **business requirement analysis phase** and involves **testing the product in user environment.**

2. Acceptance tests uncover the compatibility issues with the other systems available in the user environment.

3. It also discovers the non-functional issues such as load and performance defects in the actual user environment.

## V- Model ─ Application

* 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.

**Advantage of V-Model**

* This is a highly-disciplined model and Phases are completed one at a time.
* Works well for smaller projects where requirements are very well understood.
* 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.

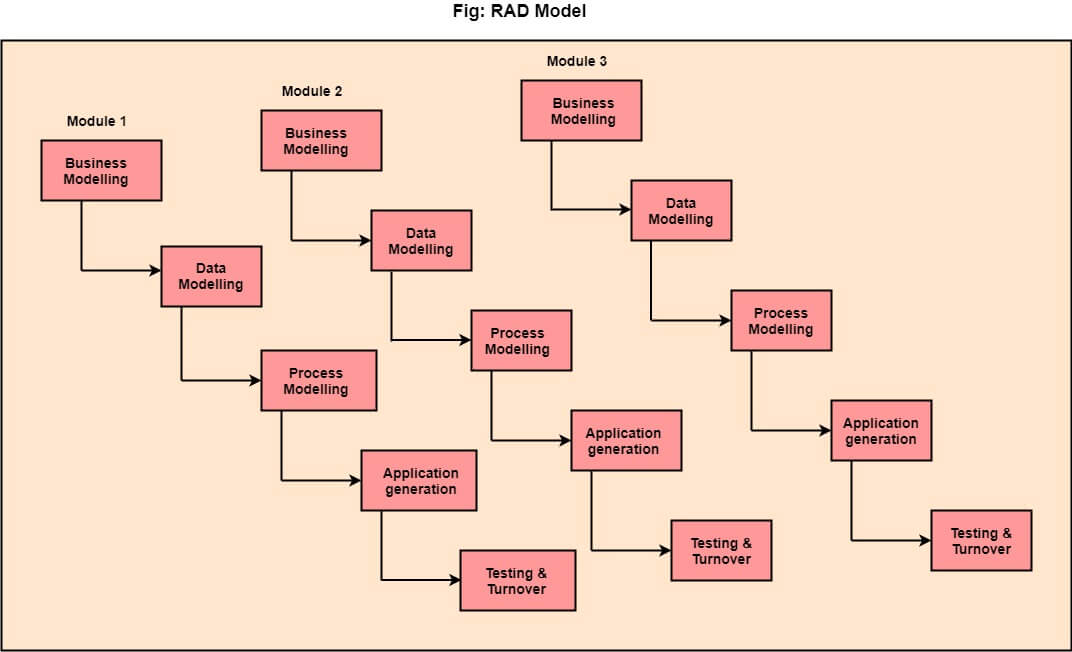
**Disadvantages of V-Model:**

* High 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.
* Once an application is in the testing stage, it is difficult to go back and change a functionality.
* No working software is produced until late during the life cycle

***5. RAPID APPLICATION DEVELOPMENT***

* Rapid Application Development (RAD) is a software development methodology that focuses on building applications in a very short amount of time
* It is a high speed adaptation of the linear sequential model in which rapid development is achieved by using component based construction.

**RAD MODEL**



**CORE ELEMENTS OF RAD**

* **Business modeling**
* **Data modeling**
* **Process modeling**
* **Application generation**
* **Testing and turnover**
* **RAD TOOLS(Erwin, CASE Tools, Rational rose,**

**Visio).**

**What mainly reduced by using RAD model?**

* **Reduced Cycle time and improved productivity**
* **How does customer involvement in the SDLC CYCLE help?**
* **Customer is involved throughout the complete cycle minimizes risk of not achieving customer satisfaction and business needs.**

**Application of RAD model**

* **RAD reasonably well known requirements.**
* **User involved throughout the life cycle.**
* **Project can be time boxed.**
* **High performance not required.**
* **Low technical risk.**
* **Functionality delivered in increments.**
* **System can be modularized and it can be broken down multiple non overlapping components.**
* **It should be used if there is a high availability of designers for Modeling.**
* **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 project and working prototypes are to be presented to customer in small iterations of 2-3 months.**

**PROS OF RAD MODEL**

* **Progress can be measured.**
* **Iteration time can be short with use of powerful RAD tools.**
* **Productivity with fewer people in a short time.**
* **Reduced development time.**
* **Increases reusability of components.**
* **Quick initial reviews occur.**
* **Encourages customer feedback.**
* **Integration from very beginning solves a lot of integration issues.**

**CONS OF RAD MODEL**

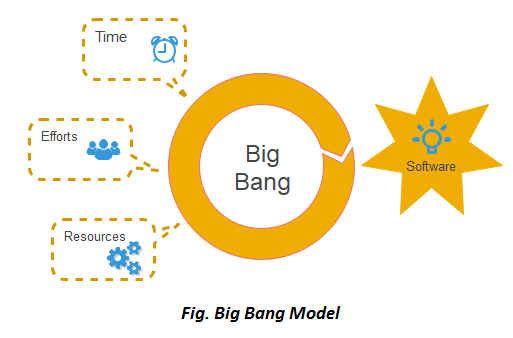
* **Dependency on technically strong team members for identifying business requirements.**
* **Only system that can be modularized can be built using RAD.**
* **Requires highly skilled developers/designers.**

***6.*** ***BIG BANK MODEL***

1. The Big Bang model is an SDLC model where we do not follow any specific process.

2. The development just starts with the required money and efforts as the input, and the output is the software developed which may or may not be as per customer requirement.

This model is ideal for small projects like academic projects or practical projects. One or two developers can work together on this model.



**When to use Big Bang Model?**

* As we discussed above, this model is required when this project is small like an academic project or a practical project.
* This method is also used when the size of the developer team is small and when requirements are not defined, and the release date is not confirmed or given by the customer.

## Big Bang Model ─ Design and Application

* The Big Bang Model comprises of focusing all the possible resources in the software development and coding, with very little or no planning.
* The requirements are understood and implemented as they come.
* Any changes required may or may not need to revamp the complete software.
* This model is ideal for small projects with one or two developers working together and is also useful for academic or practice projects.
* It is an ideal model for the product where requirements are not well understood and the final release date is not given.

# Advantages of the Big Bang Model

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

# Disadvantages of the Big Bang Model

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