**Activity 6**

**Name:Nishanthini V**

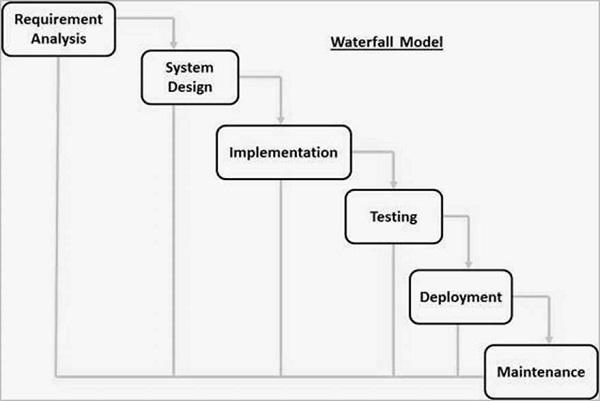
**Batch No:2022-9615**

**Enroll number: EBEON0323765163**

**SDLC MODEL**

**1)Waterfall model**

Used in software engineering and product development, the waterfall model is a linear, sequential approach to the software development lifecycle (SDLC). The waterfall approach applies a logical sequence of SDLC processes to a project in a manner akin to how waterfalls from a hill.



**Advantage**

* Simple and easy to understand and use
* Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process.
* Phases are processed and completed one at a time.
* Works well for smaller projects where requirements are very well understood.
* Clearly defined stages.
* Well understood milestones.
* Easy to arrange tasks.
* Process and results are well documented.
* It takes till the end of the life cycle for any working software to be generated.

**Disadvantage**

* Risk and uncertainty are very high.
* Unsuitable as a model for intricate and object-oriented projects.
* Ineffective model for continuing, protracted projects.
* Not appropriate for projects where there is a moderate to high probability of requirements changing. Hence, with this process model, risk and uncertainty are considerable.
* It is challenging to gauge development within stages.
* unable to adapt to changing needs.
* The project life cycle can be terminated by changing the scope.
* Integration is completed in a "big-bang" at the very end, which prevents the early detection of any business or technology bottlenecks or issues.

2**) ITERATIVE MODEL**

It is a particular implementation of a software development life cycle that focuses on an initial, simplified implementation, which then progressively gains more complexity and a broader feature set until the final system is complete. In short, iterative development is a way of breaking down the software development of a large application into smaller pieces.



**Advantage**

* Some working functionality can be developed quickly and early in the life cycle.
* Results are obtained early and periodically.
* Parallel development can be planned.
* Progress can be measured.
* Less costly to change the scope/requirements.
* Testing and debugging during smaller iteration is easy.
* Risks are identified and resolved during iteration; and each iteration is an easily managed milestone.
* Easier to manage risk - High risk part is done first.
* Risk analysis is better.
* It supports changing requirements.
* Initial Operating time is less.
* During the life cycle, software is produced early which facilitates customer evaluation and feedback**.**

**Disadvantage**

* More resources may be required.
* Although cost of change is lesser, but it is not very suitable for changing requirements.
* More management attention is required.
* System architecture or design issues may arise because not all requirements are gathered in the beginning of the entire life cycle.
* Defining increments may require definition of the complete system.
* Not suitable for smaller projects.
* Management complexity is more.
* End of project may not be known which is a risk.
* Highly skilled resources are required for risk analysis.
* Projects progress is highly dependent upon the risk analysis phase.

**3)SPRIAL MODEL**

The spiral model is based on the idea of continuously refining and improving the software product through a series of iterations. Each iteration involves a set of activities such as planning, risk analysis, engineering, and evaluation.



**Advantage**

* Changing requirements can be accommodated.
* Allows extensive use of prototypes.
* Requirements can be captured more accurately.
* Users see the system early.
* Development can be divided into smaller parts and the risky parts can be developed earlier which helps in better risk management

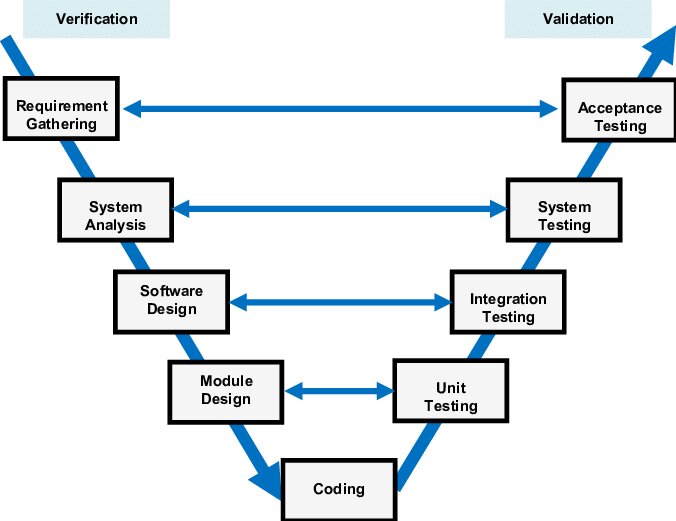
**Disadvantage**

* Management is more complex.
* End of the project may not be known early.
* Not suitable for small or low risk projects and could be expensive for small projects.
* Process is complex
* Spiral may go on indefinitely.
* Large number of intermediate stages requires excessive documentation.

**4)V MODEL**

V-Model also referred to as the Verification and Validation Model. In this, each phase of SDLC must complete before the next phase starts. It follows a sequential design process similar to the waterfall model. Testing of the device is planned in parallel with a corresponding stage of development.

V-Model contains Verification phases on one side of the Validation phases on the other side. Verification and Validation process is joined by a coding phase in V-shape. Thus it is known as V-Model.



**Advantage**

* Easy to Understand.
* Testing Methods like planning, test designing happens well before coding.
* This saves a lot of time. Hence a higher chance of success over the waterfall model.
* Avoids the downward flow of the defects.
* Works well for small plans where requirements are easily understood.

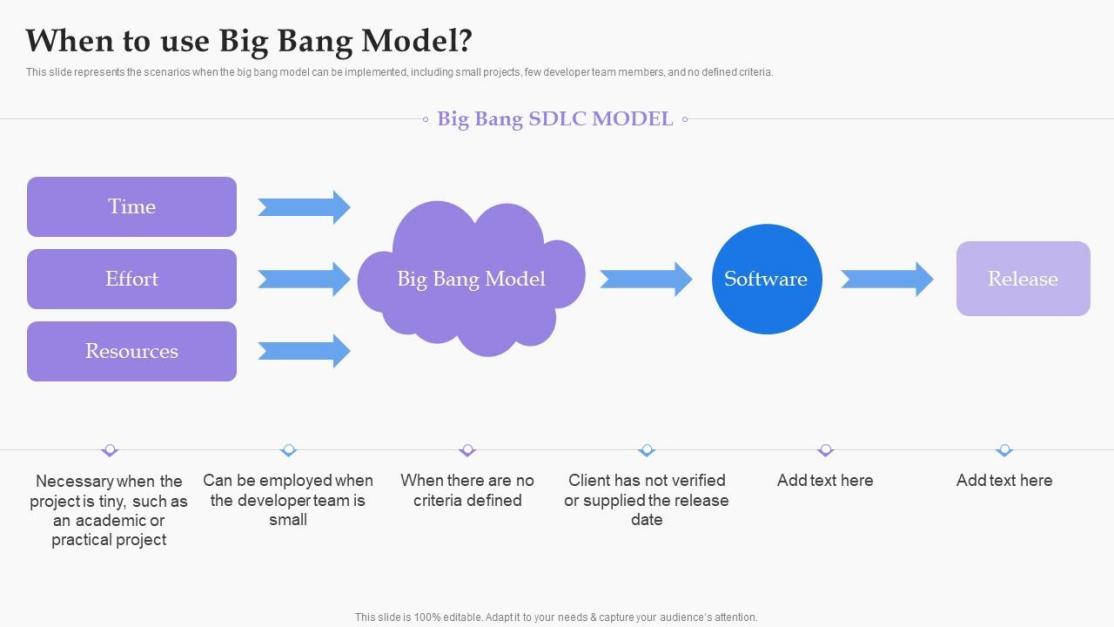
**Disadvantage**

* Very rigid and least flexible.
* Not good for a complex project.
* Software is developed during the implementation stage, so no early prototypes of the software are produced.
* If any changes happen midway, then the test documents along with the required documents, have to be updated.

**5)Big Bang MODEL**

In this model, developers do not follow any specific process. Development begins with the necessary funds and efforts in the form of inputs. And the result may or may not be as per the customer's requirement, because in this model, even the customer requirements are not defined.

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



**Advantage**

* There is no planning required.
* Simple Model.
* Few resources required.
* Easy to manage.
* Flexible for developers.

**Disadvantage**

* There is high risk and uncertainty.
* Not acceptable for a large project.
* If requirements are not clear, that can be very expensive.

**6)Agile MODEL**

The Agile model is an SDLC model focused on iterative software product

development. The Agile paradigm is an incremental model in which

software is built in quick increments. The most significant aspect of the

Agile model is determining the project scope, requirements, number, and

duration of iterations at the start of the development process.



**ADVANTAGES**

* Frequently encourages the delivery of working software.
* Adapts to changes in circumstances during the project.
* Reduces the total time spent in development.
* Constant interactions between clients, developers, and testers.

**DISADVANTAGES**

* There may be confusion and misinterpretation at any time by
* different team members due to the shortage or lack of formal documents.
* It can be difficult to maintain the final project due to the lack of
* proper documentation.
* It is difficult to assess the effort required to produce deliverables at the beginning of the software development process.

**7) RAD MODEL**

The RAD (Rapid Application Development) model is based on

prototyping and iterative development with no specific planning involved.

The process of writing the software itself involves the planning required

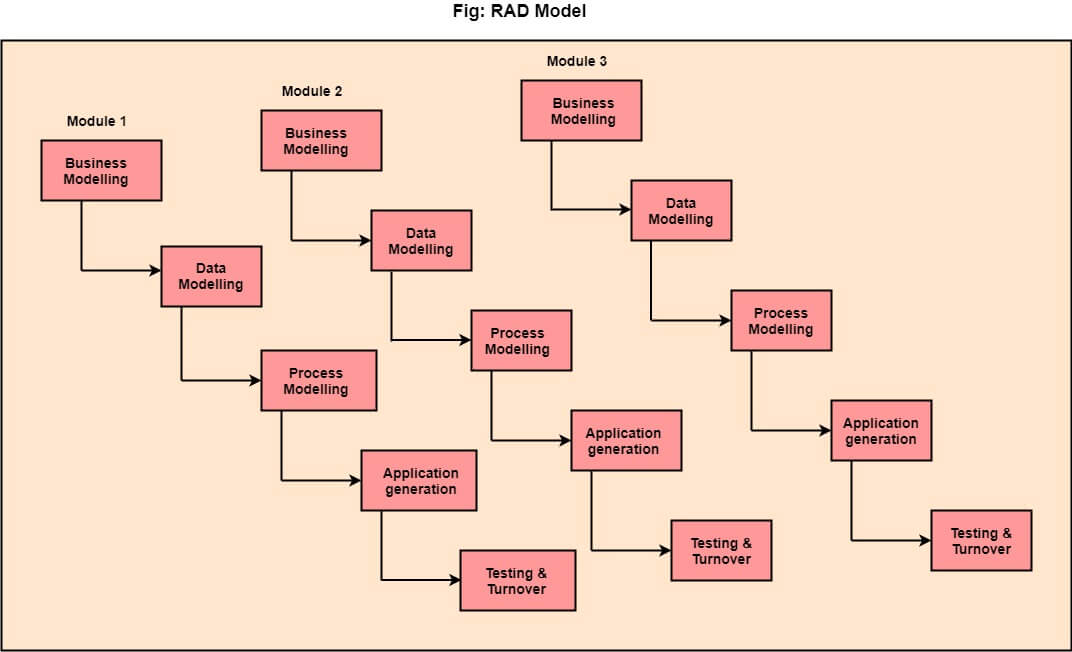
for developing the product.

Rapid Application Development focuses on gathering customer

requirements through workshops or focus groups, early testing of the

prototypes by the customer using iterative concept, reuse of the existing

prototypes (components), continuous integration and rapid delivery.



**ADVANTAGES**

* This model is flexible for change.
* In this model, changes are adoptable.
* Each phase in RAD brings highest priority functionality to the customer.
* It reduced development time.
* It increases the reusability of features.

**DISADVANTAGES**

* It required highly skilled designers.
* All application is not compatible with RAD.
* For smaller projects, we cannot use the RAD model.
* On the high technical risk, it's not suitable.
* Required user involvement.