Comparison of Software Development Life Cycle Models

Asst.Prof. Vibhuti P. Patel[1], Asst.Prof. Bhumika H. Patel[2], Asst.Prof. Tejas B. Patel[3]

[1] *Assistant Professor, Computer Engineering Department, GIDC Degree Engineering College-Abrama-Navsari, Gujarat, India*
[2] *Assistant Professor, Computer Engineering Department, GIDC Degree Engineering College-Abrama-Navsari, Gujarat, India*
[3] *Assistant Professor, Computer Engineering Department, GIDC Degree Engineering College-Abrama-Navsari, Gujarat, India*

ABSTRACT

This Paper provides you to comparison of various SDLC (Software Development Life Cycle) model of software. This paper provide you the advantage, disadvantage and limitation and also gives you best use of these models according to situation. Paper which includes Waterfall Model, Iterative water fall Model, Prototype Model, Spiral Model and Agile Model.

Keyword: - *SDLC, Waterfall, Iterative Waterfall, Prototype, Spiral, Agile*

1. INRODUCTION:

The systems development life cycle (SDLC), also referred to as the application development life-cycle, is a term used in systems engineering, information systems and software engineering to describe a process for planning, creating, testing, and deploying an information system.[1] The systems development lifecycle concept applies to a range of hardware and software configurations, as a system can be composed of hardware only, software only, or a combination of both.

Main Phases of any software to develop are:

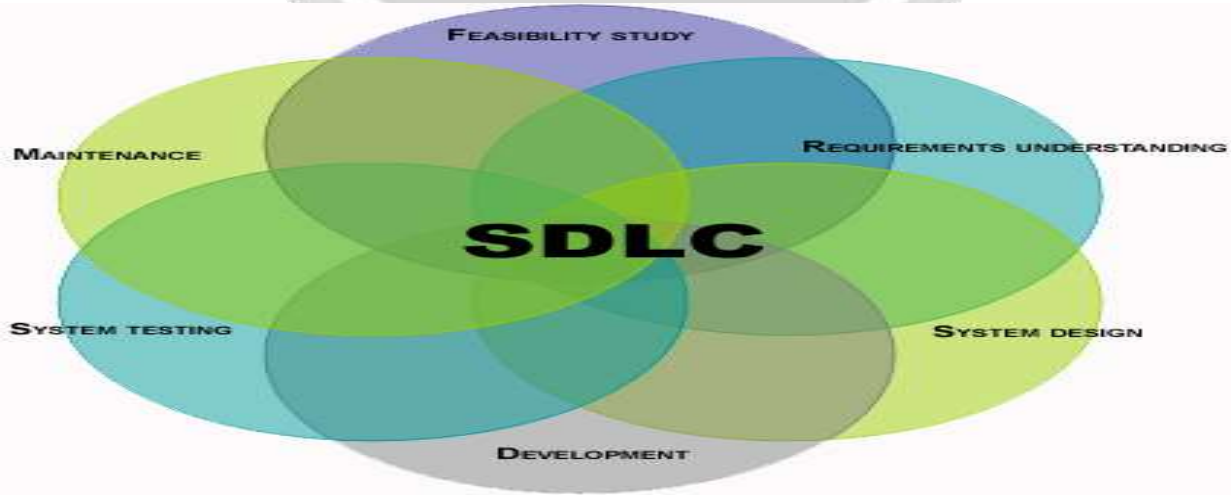

Fig-1: SDLC

1.1 Feasibility Study

The phase is to examine the **feasibility** of the system. The **feasibility study** is basically the test of the proposed system in the light of its workability, meeting user's requirements, effective use of resources and of course, the cost effectiveness od software.

1.2 Requirement understanding

The second phase in the SDLC is the requirements gathering and analysis phase. The most important phase of the SDLC is the requirement gathering and analysis phase because this is when the project team begins to understand what the customer wants from the project.

1.3 system Design

Based on the user requirements third phase is designing and the detailed analysis of a new system, the new system must be designed. It is the most crucial phase in the development of a system. The logical system design arrived at as a result of system analysis and is converted into physical system design. In the design phase the SDLC process continues to move from the what questions of the analysis phase to the how .

1.4 Development

After the designing they demands the coding of design into computer language, i.e., programming language. This is also called the programming phase in which the programmer converts the program specifications into computer instructions, which we refer to as programs. It is an important stage where the defined procedures are transformed into control specifications by the help of a computer language.

1.5 System Testing

After actual implementation the new system into operations, a test run of the system is done removing all the bugs, if any. It is an important phase of a successful system. After codifying the whole programs of the system, a test plan should be developed and run on a given set of test data. The output of the test run should match the expected results. Sometimes, system testing is considered as a part of implementation process.

1.6 Maintenance

Maintenance is necessary to eliminate errors in the system during its working life and to tune the system to any variations in its working environments. It must meet the scope of any future enhancement, future functionality and any other added functional features to cope up with the latest future needs.

2. WaterFall Model

Waterfall approach was first SDLC Model to be used widely in Software Engineering to ensure success of the project. In "The Waterfall" approach, the whole process of software development is divided into separate phases. In Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially.

Following is a diagrammatic representation of different phases of waterfall model

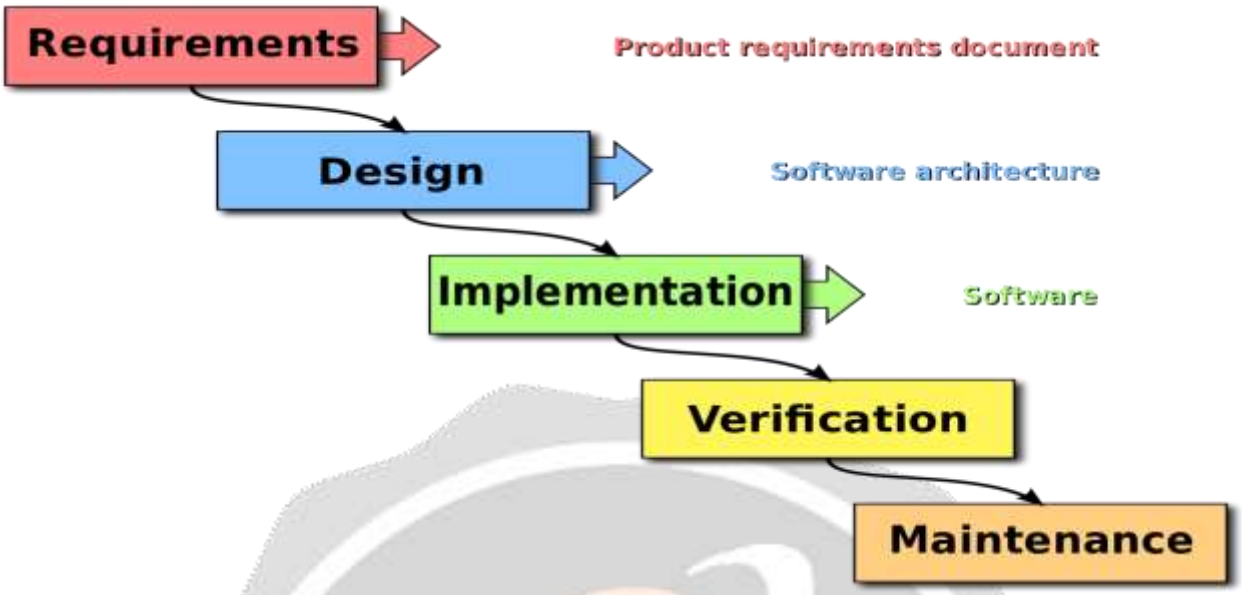

Fig-2: Waterfall Model

The sequential phases in Waterfall model are:

1. Fesibilty Study
2. Requirment gathering & Analysis
3. Designing
4. Coding
5. Testing
6. validation
7. Maintenance

3. Iterative waterfall Model

The main difference between waterfall and this model is feedback path. This model provide feedback path to correct error in any phase.

Iterative process starts with a simple implementation of a subset of the software requirements and iteratively enhances the evolving versions until the full system is implemented. At each iteration, design modifications are made and new functional capabilities are added. The basic idea behind this method is to develop a system through repeated cycles iterative and in smaller portions at a time incremental.

Following is the pictorial representation of Iterative and Incremental model:

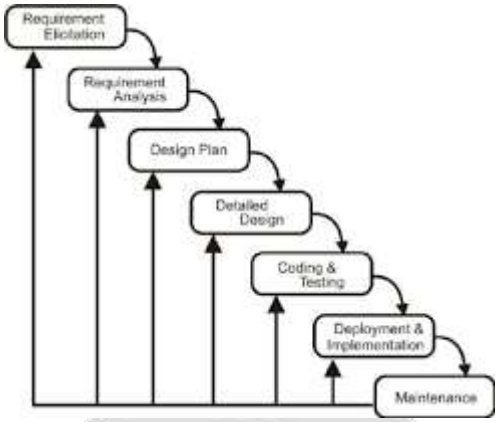

Fig-3: Iterative Waterfall model

Iterative and Incremental development is a combination of both iterative design or iterative method and incremental build model for development. "During software development, more than one iteration of the software development cycle may be in progress at the same time." and "This process may be described as an "evolutionary acquisition" or "incremental build" approach." In incremental model the whole requirement is divided into various builds. During each iteration, the development module goes through the requirements, design, implementation and testing phases. Each subsequent release of the module adds function to the previous release. The process continues till the complete system is ready as per the requirement. The key to successful use of an iterative software development lifecycle is rigorous validation of requirements, and verification & testing of each version of the software against those requirements within each cycle of the model. As the software evolves through successive cycles, tests have to be repeated and extended to verify each version of the software.

4. Prototype Model

The Software Prototyping refers to building software application prototypes which display the functionality of the product under development but may not actually hold the exact logic of the original software. Software prototyping is becoming very popular as a software development model, as it enables to understand customer requirements at an early stage of development. It helps get valuable feedback from the customer and helps software designers and developers understand about what exactly is expected from the product under development. What is Software Prototyping? Prototype is a working model of software with some limited functionality. The prototype does not always hold the exact logic used in the actual software application and is an extra effort to be considered under effort estimation. Prototyping is used to allow the users evaluate developer proposals and try them out before implementation. It also helps understand the requirements which are user specific and may not have been considered by the developer during product design.

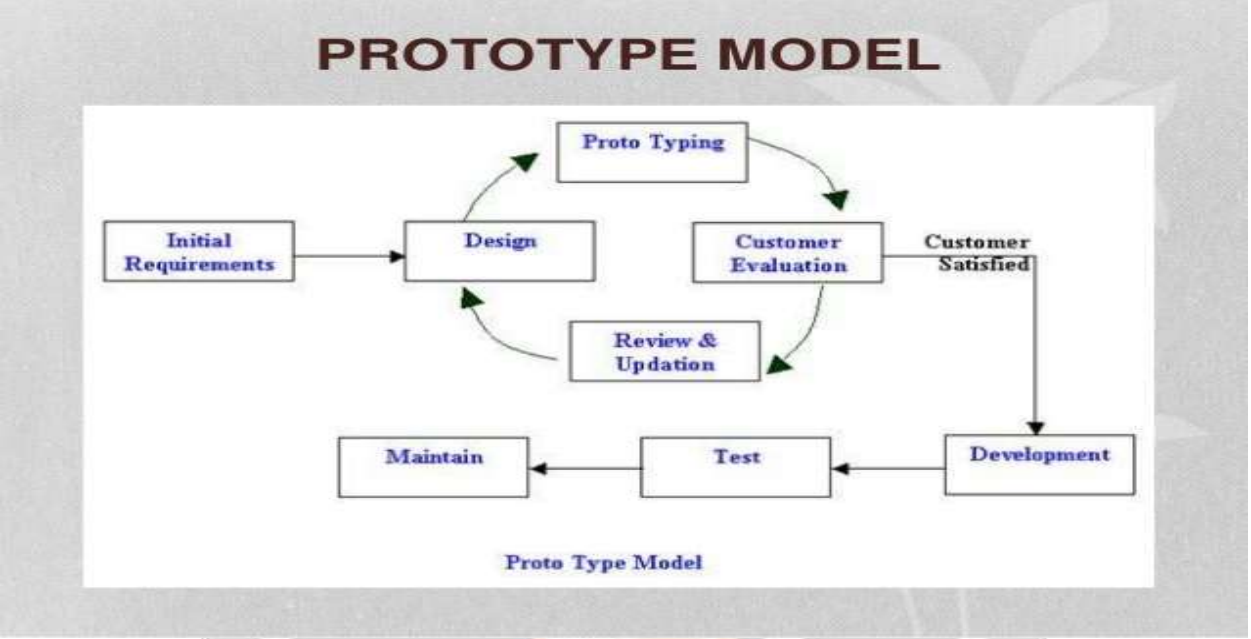

Fig-4: Prototype Model

Following is the stepwise approach to design a software prototype:

1.Basic Requirement Identification:

This step involves understanding the very basics product requirements especially in terms of user interface. The more intricate details of the internal design and external aspects like performance and security can be ignored at this stage.

2.Developing the initial Prototype:

The initial Prototype is developed in this stage, where the very basic requirements are showcased and user interfaces are provided. These features may not exactly work in the same manner internally in the actual software developed and the workarounds are used to give the same look and feel to the customer in the prototype developed.

3.Review of the Prototype:

The prototype developed is then presented to the customer and the other important stakeholders in the project. The feedback is collected in an organized manner and used for further enhancements in the product under development.

4.Revise and enhance the Prototype:

The feedback and the review comments are discussed during this stage and some negotiations happen with the customer based on factors like , time and budget constraints and technical feasibility of actual implementation. The changes accepted are again incorporated in the new Prototype developed and the cycle repeats until customer expectations are met.

5. Spiral Model

The spiral model combines the idea of iterative development with the systematic, controlled aspects of the waterfall model. Spiral model is a combination of iterative development process model and sequential linear development

model i.e. waterfall model with very high emphasis on risk analysis. It allows for incremental releases of the product, or incremental refinement through each iteration around the spiral. Spiral Model design

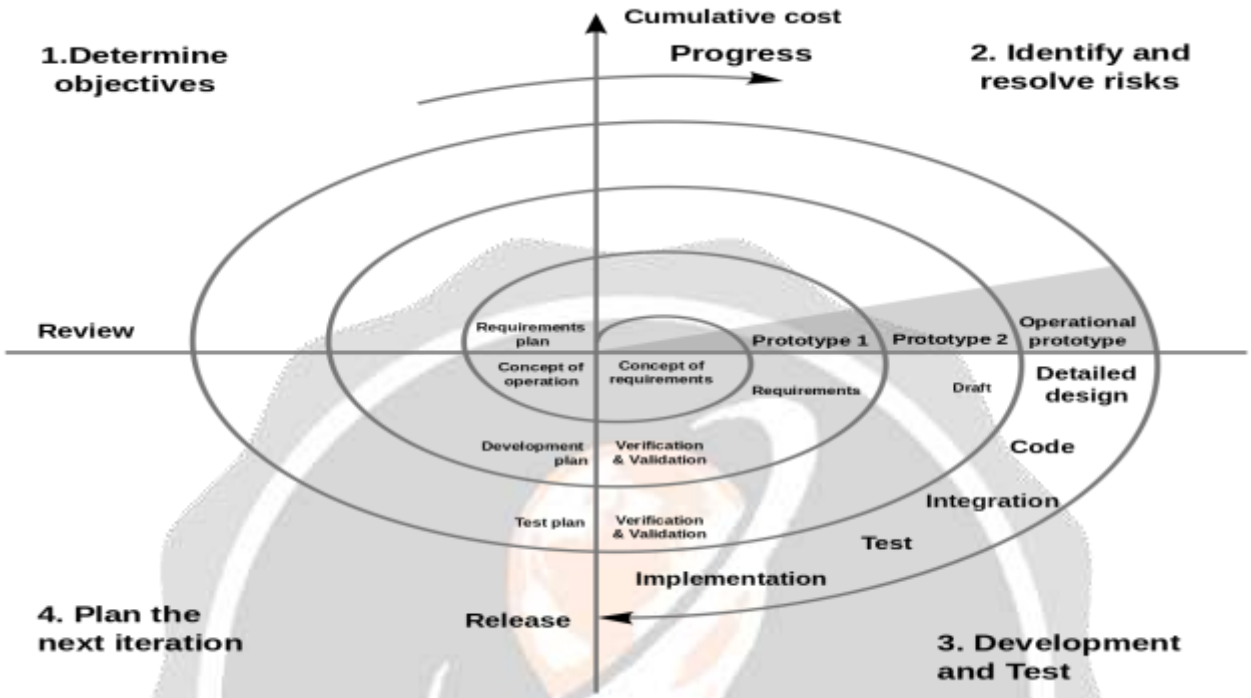

Fig-5:Spiral Model

The spiral model has four phases. A software project repeatedly passes through these phases in iterations called Spirals.

5.1 Identification:

This phase starts with gathering the business requirements in the baseline spiral. In the subsequent spirals as the product matures, identification of system requirements, subsystem requirements and unit requirements are all done in this phase. This also includes understanding the system requirements by continuous communication between the customer and the system analyst. At the end of the spiral the product is deployed in the identified market.

5.2 Design:

Design phase starts with the conceptual design in the baseline spiral and involves architectural design, logical design of modules, physical product design and final design in the subsequent spirals.

5.3 Construct or Build:

Construct phase refers to production of the actual software product at every spiral. In the baseline spiral when the product is just thought of and the design is being developed a POC ProofofConcept is developed in this phase to get customer feedback. Then in the subsequent spirals with higher clarity on requirements and design details a working model of the software called build is produced with a version number. These builds are sent to customer for feedback.

5.4 Evaluation and Risk Analysis:

Risk Analysis includes identifying, estimating, and monitoring technical feasibility and management risks, such as schedule slippage and cost overrun. After testing the build, at the end of first iteration, the customer evaluates the software and provides feedback.

6. Agile Model

Agile SDLC model is a combination of iterative and incremental process models with focus on process adaptability and customer satisfaction by rapid delivery of working software product. Agile Methods break the product into small incremental builds. These builds are provided in iterations.

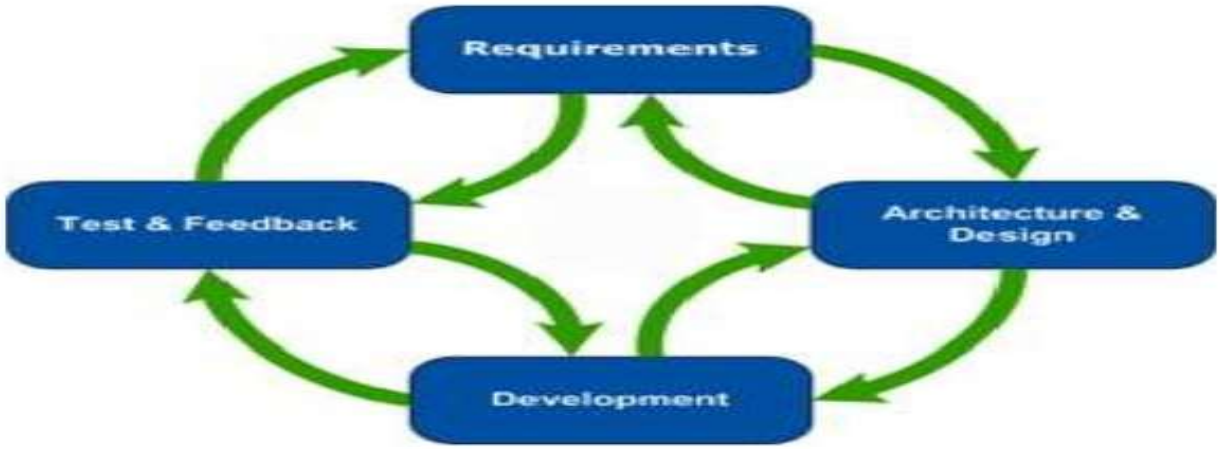

Fig-6: Agile Model

Every iteration involves cross functional teams working simultaneously on various areas like planning, requirements analysis, design, coding, unit testing, and acceptance testing. At the end of the iteration a working product is displayed to the customer and important stakeholders.

7. CONCLUSIONS

This paper shows the comparison of different model of software development life cycle to develop software. In this paper we explain 6 model with its description how to use this model. As per these comparison we conclude that Agile model is best model.

8s. REFERENCES

[1]. www.tutorialspoint.com
[2]. A detailed study of Software Development Life Cycle (SDLC) Models by Sahil Barjtya1, Ankur Sharma2, Usha Rani3in International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 6 Issue 7 July 2017, Page No. 22097-22100
[3]. www.smartsheet.com