Multiple SDLC Models With Comparison

Rizvi Khurram Abbas[1], Shaikh Mohd Ashfaque[2], Shaikh Farhan[3], Shaikh Juned[4], Mohammad Shaziuzzama[5]
[1, 2] Dept of Computer
[3,4]Dept of Electronics Engineering
[5]Dept of Humanities & Sciences
[1, 2,3, 4,5] Rizvi College Of Engineering,Mumbai, India

Abstract*- Software Development Life Cycle (SDLC) is one of the important methodologies used for the development of any software. In this paper we have mentioned some of the various methodologies that are used for software development and which methodology is suitable for any specific scenario. The software development life cycle thus plays an important role in producing high quality software that meets the goals of its development. The paper begins with the discussion to the introduction of SDLC, followed by the comparison among the various SDLC Models.*

Keywords- SDLC, Methodologies, Models.

I. INTRODUCTION

Software Development Life Cycle is a process that is used to design, develop and test desired software. The aim of SDLC is to produce high quality software that meets the user's expectation. It also ensures that the project is completed in a given time frame and budget constraints. Selecting a Software Development Life Cycle (SDLC) methodology is a challenging task for many organizations and software engineers. What tends to make it challenging is the fact that few organizations know what are the criteria to use in selecting a methodology to add value to the organization. Fewer still understand that a methodology might apply to more than one Life Cycle Model. Before considering a framework for selecting a given SDLC methodology, we need to understand the different types and the advantages and disadvantages of those models [1]. Various processes and methodologies have been developed over the last few decades to improve software quality, with varying degrees of success. However it is widely agreed that no single approach that will prevent project over runs and failures in all cases. Software projects that are large, complicated, poorly-specified, and involve unfamiliar aspects, are still particularly vulnerable to large, unanticipated problems. A software development process is a structure imposed on the development of a software product. There are several models for such processes, each describing approach test to a variety of tasks or activities that take place during the process. It aims to be the standard that defines all the tasks required for developing and maintaining software. [2]

These classic software life cycle models usually include Some version or subset of the following activities:

Planning, Analysis, Design, Implementation, Coding, Testing, Deployment and Support.

Planning
Analysis
Support
Software Development LifeCycle Model
Deployment
Design
Testing
Implement ation
Coding

fig.1: SDLC

II. VARIOUS SDLC MODELS

The various SDLC models are discussed as:

a) Waterfall model*:* The Waterfall model is a linear sequential flow. In which progress is seen as flowing steadily downwards (like a waterfall) through the phases of software implementation. This means that any phase in the development process begins only if the previous phase is complete. The waterfall approach does not define the process to go back to the previous phase to handle changes in requirement. The waterfall approach is the earliest approach and most widely known that was used for software development. [3]

Waterfall model is most appropriate when –

- 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.
- The project is short.

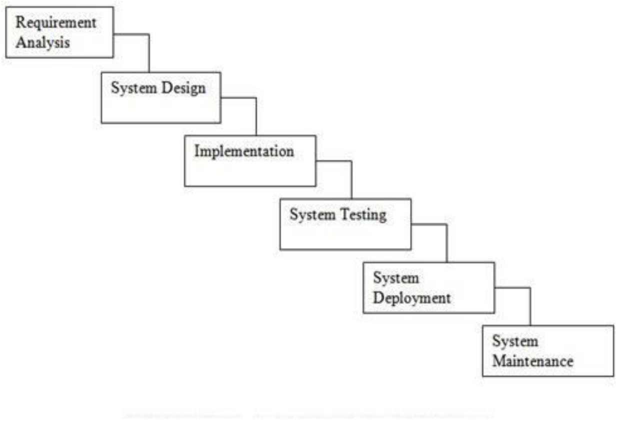


fig.2: Waterfall model

b) V-shaped 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 and is based on the association of a testing phase for each corresponding development stage [4].

The following pointers are some of the most suitable scenarios to use the 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.

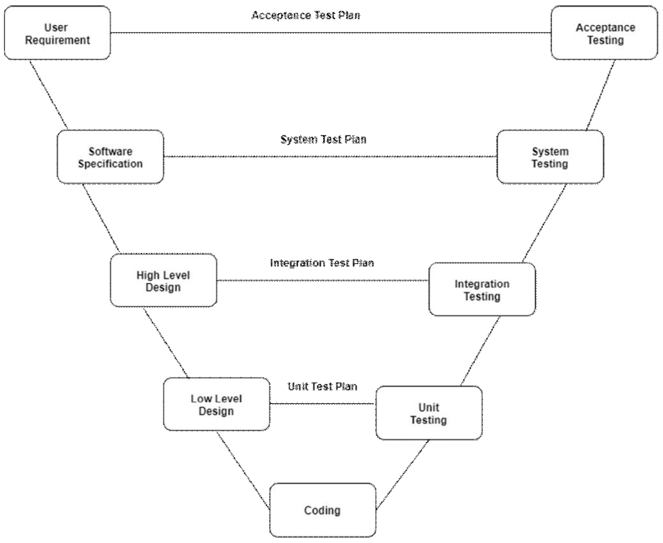


fig.3: V-shaped model

c) Prototyping model: This model is referred to the activity of creating prototype of software applications; it is an activity that can occur in software development. It used to visualize some component of the software to limit the gap of misunderstanding the customer requirements by the development team. This also will reduce the iterations that may occur in waterfall approach and hard to be implemented due to the inflexibility of the waterfall approach [5].

Prototypes can have horizontal or vertical dimensions. A Horizontal prototype displays the user interface for the product and gives a broader view of the entire system, without concentrating on internal functions. A Vertical prototype on the other side is a detailed elaboration of a specific function or a sub system in the product [5].

The types of prototyping models are:

- **Throwaway prototyping:** Prototypes that are eventually discarded rather than becoming a part of the finally delivered software [5].

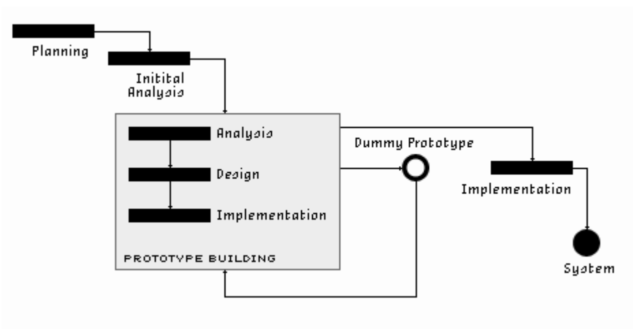


fig.4 (a): Throwaway prototyping model

- **Evolutionary prototyping:** prototypes that evolve into the final system through an iterative incorporation of user feedback [5].

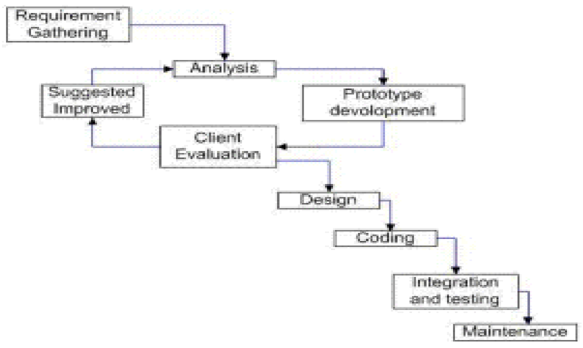


fig.4 (b): Evolutionary Prototyping model

- **Incremental prototyping:** The final product is built as separate prototypes. At the end, the separate prototypes are merged in an overall design [5].

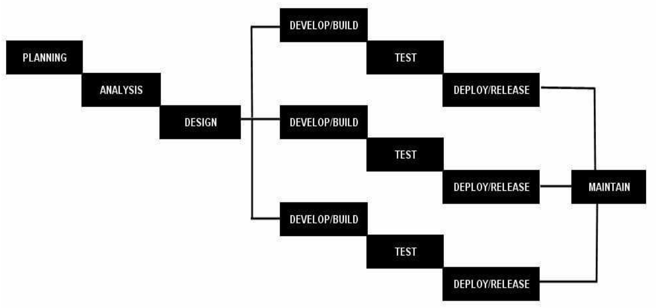


fig.4 (c): Incremental prototyping model

- **Extreme prototyping:** It issued in web applications mainly. Basically, it breaks down web development into three phases, each one based on the preceding one [6].

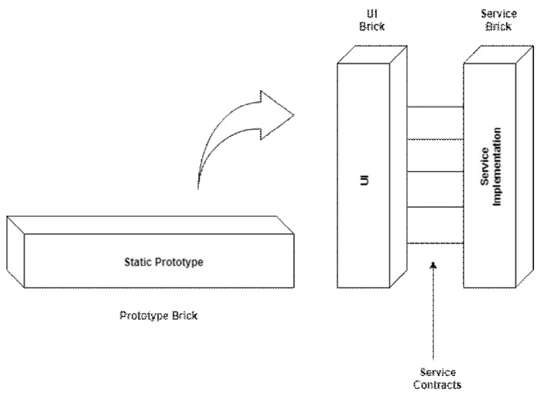


Fig.4 (d): Extreme prototyping model

d) Spiral model: The spiral model combines the idea of iterative development with the systematic, controlled aspects of the waterfall model. This Spiral model is a combination of iterative development process model and sequential linear development model i.e. the waterfall model with a very high emphasis on risk analysis. It allows incremental releases of the product or incremental refinement through each iteration around the spiral [7].

The spiral model has four phases.

- Identification
- Design
- Construct or Build
- Evaluation and Risk Analysis

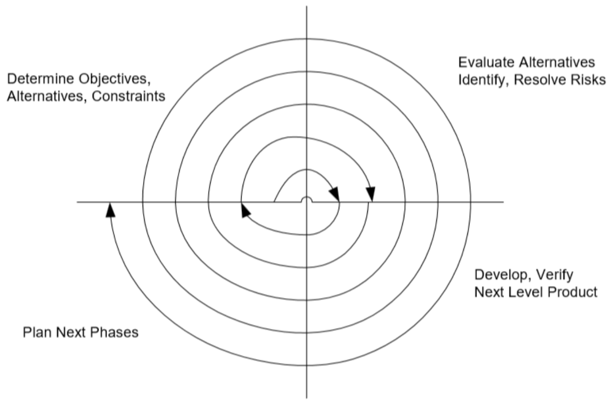

Determine Objectives, Alternatives, Constraints
Evaluate Alternatives Identify, Resolve Risks
Develop, Verify Next Level Product
Plan Next Phases

fig.5: Spiral model

The Spiral Model is widely used in the software industry as it is in sync with the natural development process of any product, i.e. learning with maturity which involves minimum risk for the customer as well as the development firms [7].

e) Iterative and Incremental model*:*In the Iterative model, iterative process starts with a simple implementation of a small set of the software requirements and iteratively enhances the evolving versions until the complete system is implemented and ready to be deployed.

An iterative life cycle model does not attempt to start with a full specification of requirements. Instead, development begins by specifying and implementing just part of the software, which is then reviewed to identify further requirements. This process is then repeated, producing a new version of the software at the end of each iteration of the model [8].

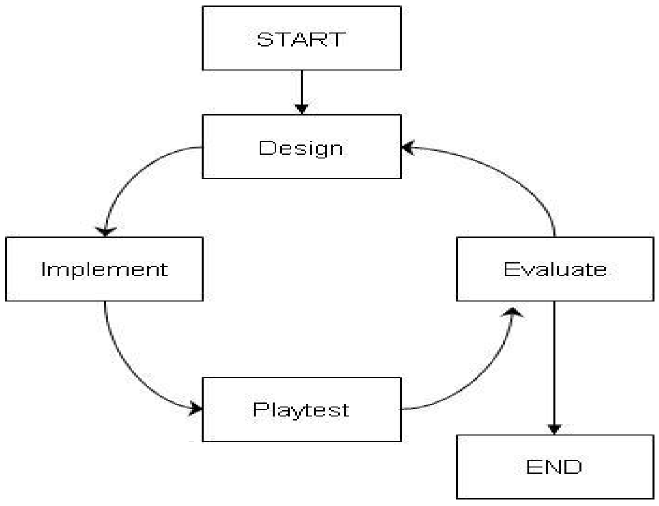


fig.5: Iterative model

This model is most often used in the following scenarios –

- Requirements of the complete system are clearly defined and understood.
- Major requirements must be defined; however, some functionalities or requested enhancements may evolve with time.
- There is a time to the market constraint.
- A new technology is being used and is being learnt by the development team while working on the project.
- Resources with needed skill sets are not available and are planned to be used on contract basis for specific iterations.
- There are some high-risk features and goals which may change in the future.

f) 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. Each iteration typically lasts from about one to three weeks [9].

Every iteration involves cross functional teams working simultaneously on various areas like

- Planning
- Requirements Analysis
- Design
- Coding
- Unit Testing and Acceptance Testing.

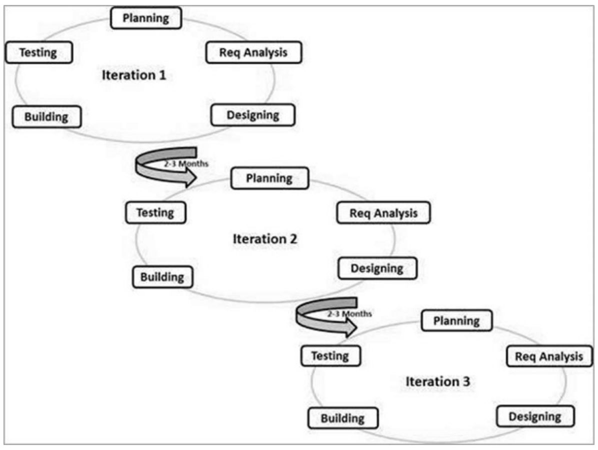


fig.6: Agile model

It can be used with any type of the project, but it needs more engagement from the customer and to be interactive. Also, it can be used when the customer needs to have some functional requirement ready in less than three weeks and the requirements are not clear enough.

g) 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, continuous integration and rapid delivery.RAD model distributes the analysis, design, build and test phases into a series of short, iterative development cycles [10].

Following are the various phases of the RAD Model

- Business modelling
- Data modelling
- Process modelling
- Application Generation
- Testing and Turnover

RAD model can be applied successfully to the projects in which clear modularization is possible.

h) Big Bang model*:* The Big Bang model is an SDLC model where we do not follow any specific process. 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.

The Big Bang Model does not follow any requirement gathering and there is very little planning required. Even the customer is not sure about what exactly he wants and the requirements are implemented on the fly without much analysis [11].

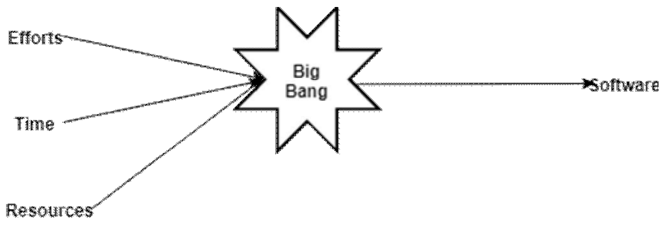
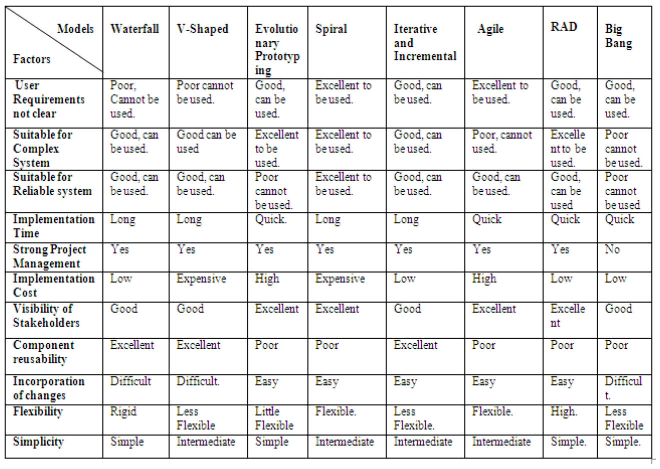


Fig.8: Big Bang model

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.

III. COMPARISON BETWEEN VARIOUS SDLCS [12, 13]



Factors \ Models	Waterfall	V-Shaped	Evolutionary Prototyping	Spiral	Iterative and Incremental	Agile	RAD	Big Bang
User Requirements not clear	Poor, Cannot be used.	Poor cannot be used.	Good, can be used.	Excellent to be used.	Good, can be used.	Excellent to be used.	Good, can be used.	Good, can be used.
Suitable for Complex System	Good, can be used.	Good can be used	Excellent to be used.	Excellent to be used.	Good, can be used.	Poor, cannot used.	Excellent to be used.	Poor cannot be used.
Suitable for Reliable system	Good, can be used.	Good, can be used.	Poor cannot be used.	Excellent to be used.	Good, can be used.	Good, can be used.	Good, can be used	Poor cannot be used
Implementation Time	Long	Long	Quick.	Long	Long	Quick	Quick	Quick
Strong Project Management	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
Implementation Cost	Low	Expensive	High	Expensive	Low	High	Low	Low
Visibility of Stakeholders	Good	Good	Excellent	Excellent	Good	Excellent	Excellent	Good
Component reusability	Excellent	Excellent	Poor	Poor	Excellent	Poor	Poor	Poor
Incorporation of changes	Difficult	Difficult.	Easy	Easy	Easy	Easy	Easy	Difficult.
Flexibility	Rigid	Less Flexible	Little Flexible	Flexible.	Less Flexible.	Flexible.	High.	Less Flexible
Simplicity	Simple	Intermediate	Simple	Intermediate	Intermediate	Intermediate	Simple.	Simple.

IV. CONCLUSION

There are various types of software development models available such as Waterfall model, V-shaped model, RAD etc. All these models have their pros and cons associated with them; however it is the responsibility of software development team to decide which model to choose based on the project requirement and size. Hence, in this research we have tried to briefly explain these models and also tried to identify their suitability in any given context.

REFERENCES

[1] SDLC and methodologies: https://melsatar.blog/2012/03/15/software-development-life-cycle-models-and-methodologies/ http://www.sdlc.ws https://www.tutorialspoint.com/sdlc/sdlc_overview.htm
[2] IJCSIT Vol. 6(1), 2015, 168-172 - ijcsit2015060137
[3] Waterfall Model https://www.tutorialspoint.com/sdlc/sdlc_waterfall_model.htm
[4] SDLC - V-Model https://www.tutorialspoint.com/sdlc/sdlc_v_model.htm
[5] SDLC - Software Prototype Model https://www.tutorialspoint.com/sdlc/sdlc_software_prototyping
[6] Extreme Prototyping https://en.wikipedia.org/wiki/Software_prototyping
[7] SDLC - Spiral Model https://www.tutorialspoint.com/sdlc/sdlc_spiral_model.htm
[8] SDLC – Incremental and Iterative Model https://www.tutorialspoint.com/sdlc/sdlc_iterative_model.htm
[9] SDLC - Agile Model https://www.tutorialspoint.com/sdlc/sdlc_agile_model.htm
[10] SDLC - RAD Model https://www.tutorialspoint.com/sdlc/sdlc_rad_model.htm
[11] SDLC - Big Bang Model https://www.tutorialspoint.com/sdlc/sdlc_bigbang_model.htm
[12] SDLC Comparison https://melsatar.blog/2012/03/21/choosing-the-right-software-development-life-cycle-model/
[13] Comparative study of various SDLC Models- IJER_2015_405
(ISSN: 2319-6890 (online), 2347-5013(print) 01 April 2015)
Big Bang Model
https://airbrake.io/blog/sdlc/big-bang-model