www.ijecs.in
International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 6 Issue 7 July 2017, Page No. 22097-22100
Index Copernicus value (2015): 58.10 DOI: 10.18535/ijecs/v6i7.32

A detailed study of Software Development Life Cycle (SDLC) Models

Sahil Barjtya[1], Ankur Sharma[2], Usha Rani[3]

[1] Arni University Dept. of Computer Science Engineering
Kangra Himachal Pradesh, India
e-mail: barjtya@gmail.com
[2]Arni University Dept. of Computer Science Engineering
Kangra Himachal Pradesh, India
e-mail: ankur.sharma.ind@gmail.com
[3]Arni University Dept. of Computer Science Engineering
Kangra Himachal Pradesh, India
e-mail: usha.arya90@gmail.com

Abstract: ***This paper provides you comparative study of all the SDLC models and other hybrid methodologies of software development. This paper provide you all the advantages and disadvantages of the existing model and their limitation and also describe best uses of these models according to the situation. We described both contemporary models and traditional models which includes Sprial Model, Incremental Model, Spiral Model and V-Shaped Models are traditional models and Rapid Application Development Model, Agile software development comes under the contemporary models category.***

Keywords: SDLC Models, Waterfall Model, Spiral Models and Agile**.**

1. Introduction

Software Development Life cycle (SDLC) is the collection of various steps which followed for the systematic development, design and maintenance of the software projects and ensure that all the user requirement is fulfilled with least amount of resource consumption [1]. These methodologies help us delivering quality product on the time and as per the client requirement. These SDLC model is suitable for specific kind of projects we cannot deploy one single model for all the software projects because every project having different requirement that's why we always collect user requirement before we select any kind of SDLC model for the project.
We find out that our traditional models such as Waterfall, Spiral, Incremental and RAD is not able to fulfill clients satisfaction level so we move a head towards hybrid SDLC model development such as Agile process is itself a software development process [2]. Agile process is an iterative approach in which customer satisfaction is at highest priority as the customer has direct involvement in evaluating the software [3].

2. Phases of SDLC

SDLC stands for Software development life cycle. It is a consist of various phases which describes how to develop, design and maintain the software project ensuring that all the functional & user requirement, goals and objective are met. This helps in quality production and the customer satisfaction. In specific terms that are relevant to SDLC, since SDLC, or Systems Development Life Cycle, is a cyclical methodology, phases repeat, so changes can be made to the design in the next cycle.

1. Requirement Analysis

In the requirement analysis phase of SDLC (Software Development Life Cycle) where the discuss with client about his needs regarding software development. The aim of this phase to grab out all the details of the project or we can say that requirement analysis phase is to capture the detail of each requirement and to make sure everyone understands the scope of the work and how each requirement is going to be fulfilled.

2. Design

The next stage of Software Development Life Cycle is the Design phase. During the design phase, developers and technical architects start the high-level design of the software and system to be able to deliver each requirement. The technical details of the design is discussed with the stakeholders and various parameters such as risks, technologies to be used, capability of the team, project constraints, time and budget are reviewed and then the best design approach is selected for the product.

DOI: 10.18535/ijecs/v6i7.32

Figure 1: SDLC phases

3. Implementation

This is the phase where we actually implement all the requirements which are gathered from the client. In these phases coding is started as per the requirement of the client. In this phase every one start doing their work database administrator start making database programmers start coding the function or we can say modules' of the projects and front end developer stat developing an interactive GUI as per the requirement of the software.

4. Testing

Testing is the last phase of the Software Development Life Cycle before the software is delivered to customers. In this phase we check that our software is working as per our expectation or not. We also check SRS that software full fill the entire requirement that mentioned by the client at the time of agreement.

5. Deployment and Maintenance

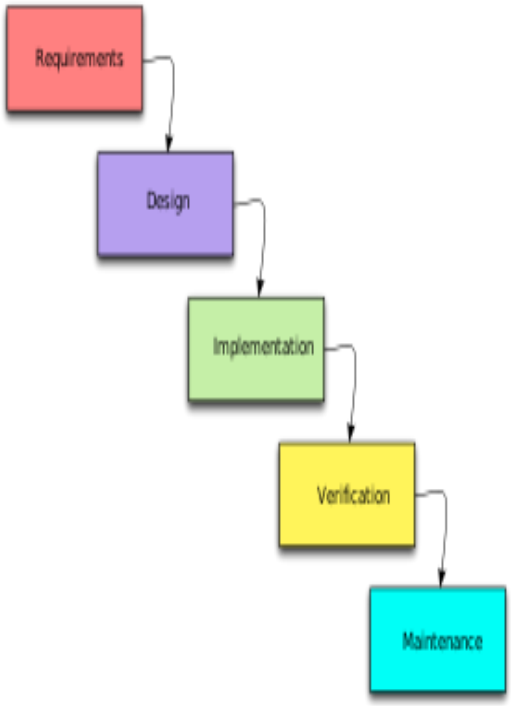
Once software development is completed we can deploy the software according to client use and we can provide there is usually a maintenance team that look after any post-production issues. If an issue is encountered in the production the development team is informed and depending on how severe the issue is, it might either require a hot-fix which is created and shipped in a short period of time or if not very severe, it can wait until the next version of the software.

3. SDLC MODELS

1. Waterfall Model

Waterfall is the traditional model of the SDLC (Software Development Lifecycle). In this model each phase is completed before going to next phase. There is no option for going back after moving to next phase. In waterfall model next phase is dependence on the result of the previous frame.

Waterfall is easy manageable and simple to understand. However, in some situation it causes to delay in project completion because prior to moving next phase we need to complete first phase. Also, since there is little room for revisions once a stage is completed, problems can't be fixed until you get to the maintenance stage.



The biggest disadvantages of this model the requirement are clear prior to project development because no client intervention is allowed in between the project. Therefore if a requirement is wrong or missing, it won't become apparent until the late stages of the life cycle. These points explain advantages and disadvantages:-

- Easy to understand
- Prevention of error propagation with the help of verification and validation
- Well defined stages
- We cannot go back to previous phase.
- Less client involvement

2.V Model

V Model is advance waterfall model in which testing functionality is added at each stage of the project development instead of the project completion project which leads to better project development. In this model also we cannot move to next step until or unless we cannot complete the previous step. In this model we not get deviated from the project goal due to each phase testing.

DOI: 10.18535/ijecs/v6i7.32

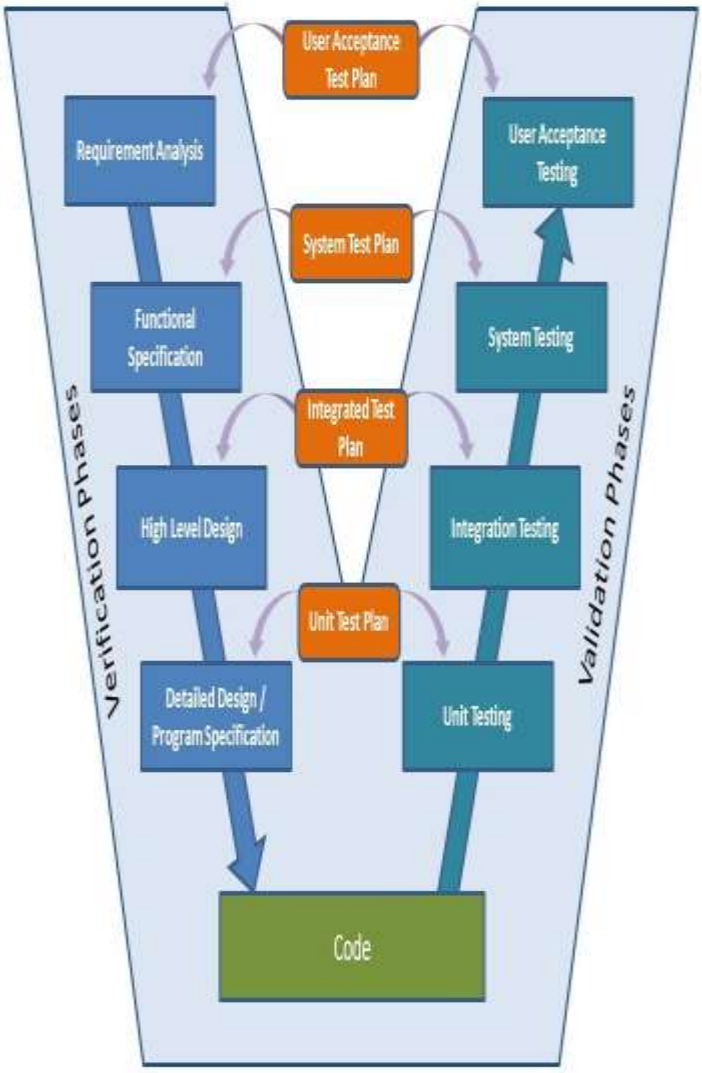


Figure: V-Shaped Model [4]

- Easy to understand and implement
- Early stage error removal
- High success rate as compare to Waterfall due to each phase testing
- Mitigate downwards flow of error.
- Not flexible and rigid model
- Highly risk is associated with this model
- Goal is not clear in this model

3.Iterative Model

With the Iterative model, the project can be developed in small chunks, each updated chunk contain some addition functionalities. In this model no need of full requirement unlike V-shaped and Waterfall model prior to stat development of the software. With each iteration, some additional requirement are added and makes an updated version of the software and this process continues until full project not get developed One advantage of Iterative model over the other SDLC methodologies is that we get a working version of the application early in the process and so it less expensive to implement changes. One disadvantage is that resources can quickly be eaten up by repeating the process again and again.

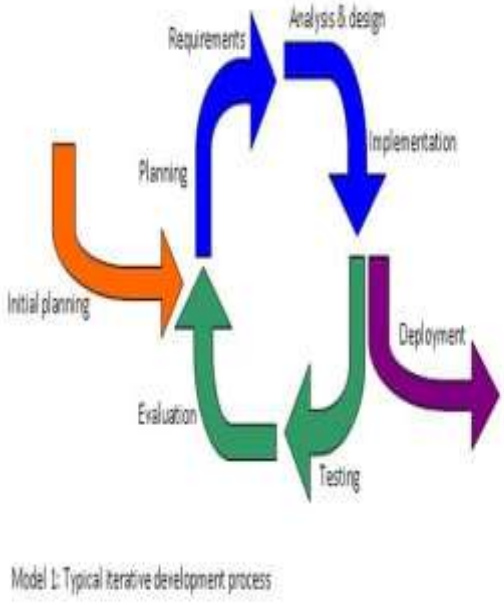


Figure: Iterative Model [5]

4.Spiral Model

Spiral model is combination of the systematic and structured development which takes attributes of iteration Iterative model and also combined these advantages with the simplicity of the waterfall model with an additional heavy risk analysis features. Working of the Spiral model is divided into four phases (identification, design, build, evaluation and risk analysis) and these four steps are get repeated until we will not get complete project. This model provides incremental updating on the releases of the software products.

Spiral model is best suited for the highly personalized software product because in this model user interaction and evaluation is started from the early stage of the development But the risk you run is creating a never-ending spiral for a project that goes on and on.

- Risk analysis is very high in this model
- Early production of software in the life cycle.
- Suitable for large projects
- Very less chance of failure
- Development can be terminated after any spiral and there will be working system available.

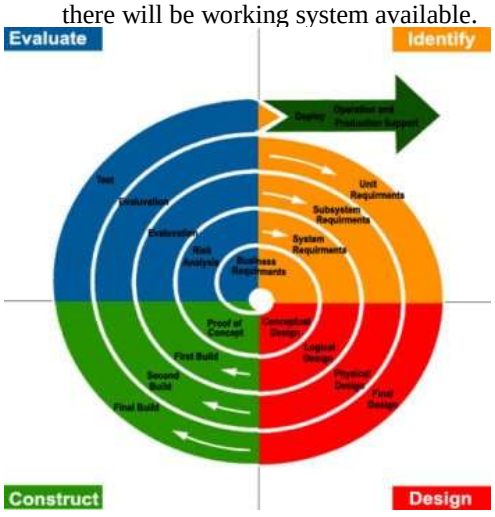


Figure: Spiral Mode [6]

5.Agile Model

The agile model is hybrid model it is uses advantages of the both iterative and incremental model by dividing software product breaking a product into apparatus where on each cycle or iteration, a working model of a component is delivered. This model delivers updated releases and each release contains some incremental updates and after completion of each iteration product is tested to ensure that the iteration is acceptable or not. The Agile model emphasizes association, as the clients, developers and testers effort mutually all through the project. An benefit of the Agile model is that it rapidly deliver a operational product and is measured a very practical development approach. One drawback of this model is that because it depends profoundly on client communication, the project can head the incorrect way if the client is not clear about the needs or the direction he or she wants to go.

- This model is very adaptable to changing requirements
- Very munch focused on client feedback
- Dynamic measure of progress
- Overhead is reduced as compare to other model
- Quick removal of horrific designs and erroneous requirements identified and removed immediately.
- Not feasible for complex project
- Agile model works well for small team
- Small projects that are developed by small, self-organizing teams
- Agile model adapt frequent changes in the technology

4. **Conclusion**

In this paper we provide brief discussion about the various Software Development Life Cycle (SDLC) models such as waterfall Model, V-shape Model, Spiral model and agile software development methodologies. We also provide advantages and shortcoming of these models with detailed

expiation of the working of these models. This paper helps you in understanding working of all SDLC models and provides deep insights about these models. In the comparative study of agile software development with other software

Development models we conclude that agile project is much better than other software development process inters of productivity, performance, faster time cycles, risk analysis

References

[1] Lehman, Tobin J., and Akhilesh Sharma. "Software development as a service: agile experiences." *SRII Global Conference (SRII), 2011 Annual*. IEEE, 2011.

[2] Ahmed, A., et al. "Agile software development: Impact on productivity and quality." *Management of Innovation and Technology (ICMIT), 2010 IEEE International Conference on*. IEEE, 2010.

[3] Boehm, Barry, and Richard Turner. Balancing Agility and Discipline: A Guide for the Perplexed, Portable Documents. Addison-Wesley Professional, 2003.

[4] https://www.roberthalf.com/technology/blog/6-basic-sdlc-methodologies-the-pros-and-cons

[5] https://www.testingexcellence.com/iterative-model/

[6] https://www.testingexcellence.com/spiral-model-sdlc/

[7] https://www.smartsheet.com/agile-vs-scrum-vs-waterfall-vs-kanban