International Journal of Engineering Research
Volume No.4, Issue No.4, pp : 188-191
ISSN:2319-6890)(online),2347-5013(print)
01 April 2015

Comparative Study of Various SDLC Models on Different Parameters

Prateek Sharma[1], Dhananjaya Singh[2]
[1] Department of BCA, BIT, Meerut-250103, India
[2] Department of Computer Science and Engineering, AIACT&R-110031, Delhi-110031, India
[1]Psharma524@gmail.com , [2]Dhananjaya1987@gmail.com

Abstract: The success of a software development project greatly depends upon which process model is used. This paper emphasizes on the need of using appropriate model as per the application to be developed. In this paper we have done the comparative study of the following software models namely Waterfall, Prototype, RAD (Rapid Application Development) Incremental, Spiral, Build and Fix and V-shaped. Our aim is to create reliable and cost effective software and these models provide us a way to develop them. The main objective of this research is to represent different models of software development and make a comparison between them to show the features of each model.

Keywords: SDLC, RAD, Adhoc, Risk Analysis, Verification and Validation, Comparative analysis of SDLC models.

I. INTRODUCTION

The need and importance of computers have grown to a very large extent in today world. Computers are being used in many areas like in banking, education, medicine etc. These areas require specialized software according to the applications they need. Hardware alone is not sufficient to do some useful work. Software and hardware are complementary to each other.

Software engineering**[6]** is an engineering discipline whose aim is development of quality product, a product which is reliable, within estimated budget and within a given time framework. Development of

sound software projects requires a proper process to be followed by the organization. This software process which is required to produce software differs from company to company. Software lifecycle model provides a method for developing a software product. A proper software life cycle model can help an organization not only in building a software product but it also serves as a basis for planning, organizing, staffing, coordinating and directing various other software development activities. Software process is properly written out and managed in organizations which are mature while organizations which are immature there is no provision for writing the software process properly. There are various software Engineering models. The adoptability of them depends upon project requirements.

In IEEE standard Glossary of Software Engineering Terminology, the software Life Cycle is: "The period of time that starts when a software product is conceived and ends when the product is no longer available for use". The software life cycle typically include the following activities which are as follows:

- Requirements Analysis
- Specification
- Software architecture
  - Implementation
  - Testing
  - Documentation
  - Training and Support
  - Maintenance

The above activities form part of framework activities and are performed in every software development project.

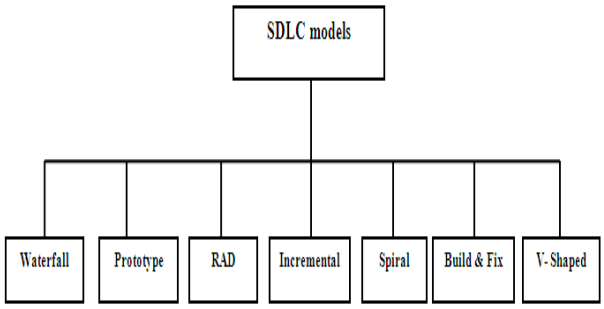

Figure1: Various SDLC Models

II. SOFTWARE PROCESS MODELS

General software process models are

1 Waterfall model

2 Prototype model

3 Rapid application development model (RAD)

4 Incremental model

5 Spiral model

6 Build and fix model

7 V-shaped model

III. SEVEN MODELS

3.1 Waterfall Model

This model is named "Waterfall" because its diagrammatic representation looks like a cascade (flow) of Waterfall. This is also known as classical lifecycle model. This is one of the oldest process model defined by Rocye**[9]** in 1970.The model begins with requirement analysis and continues with design, coding and testing. All the phases of waterfall model are independent of each other and developer must complete each phase before the next phase could begin.

ISSN:2319-6890)(online),2347-5013(print)
01 April 2015

This model is suitable for projects in which we have well known requirements and they are well defined in earlier stages. This model is simple to understand and use. It follows a sequential approach. We cannot deliver the product to the client until the final stage is completed.

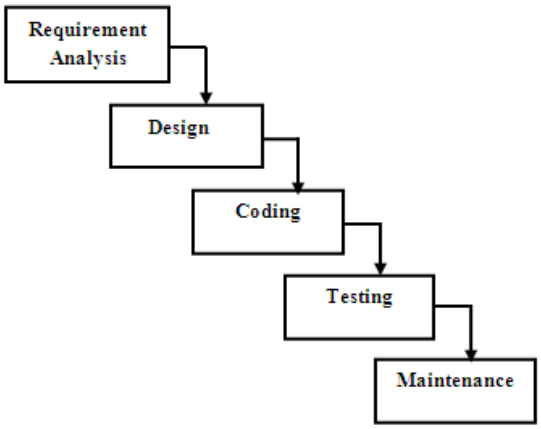

Figure2: Waterfall Model

3.2 Prototype Model

A prototype**[5]** is a toy implementation of a system. Prototyping can be evolutionary or throwaway. In prototyping first requirements of customer are gathered from them and then a working prototype is developed as per their requirements. After the prototype is developed it is provided to the customer and customer review it to see that whether it meets its expected requirements. Prototype is developed to determine the actual need of the customer. After finalization of software requirement specification developer attempts to use existing program segments from prototype and actual system is then developed using waterfall approach to produce good quality software product.

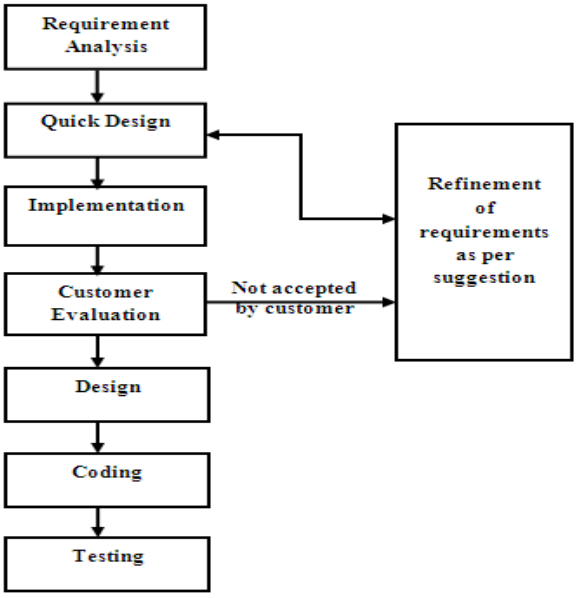

Figure3: Prototype Model

3.3 RAD Model

The full form for RAD is Rapid Application Development. RAD model is a high speed adaption of waterfall model. This model can be implemented if a developer knows the requirements of customer in advance and here the development cycle is extremely small. User or customer involvement is there in every stage of RAD model. This model has four phase requirement planning, user description, construction and cutover. A number of teams work on a single function and then it is integrated to form whole software.

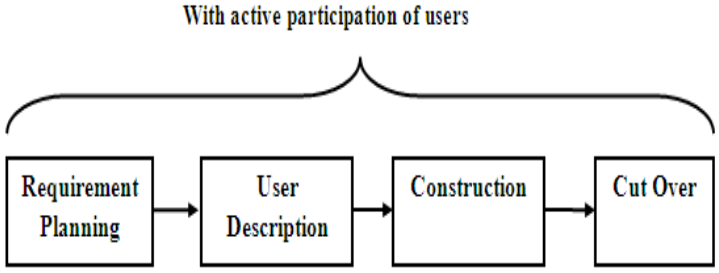

Figure4: RAD Model

3.4 Incremental Model

If a customer requires changes in its product, then incremental model**[8]** accommodate changes as required by the customer. The previous models discussed earlier do not take into consideration changes in product. This model is iterative in nature. A reusable product is released at the end of each cycle, with each release providing additional functionality. After each release customer can do some useful work and thereby he can accommodate changes in the product. The waterfall and prototype model delivers a complete operational product while iterative model delivers an operational quality product at each release.

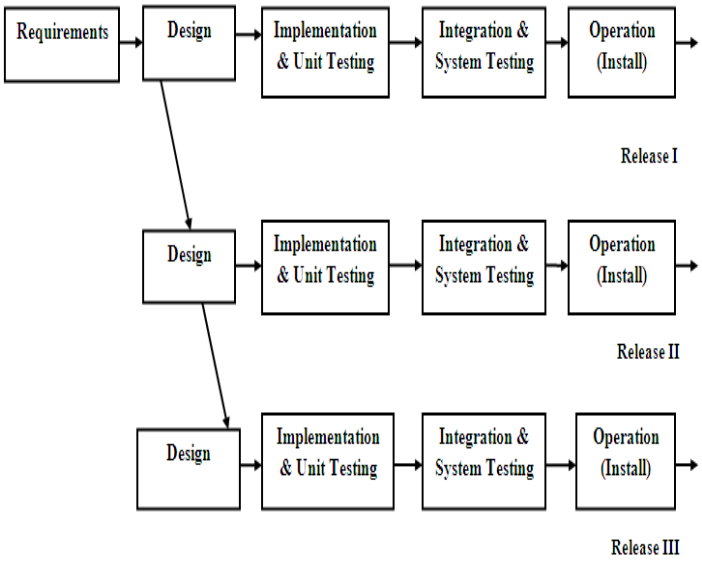

Figure5: Incremental Model

3.5 Spiral Model

Spiral model**[1]** takes into account the factor of risk analysis. The spiral model is cyclic in nature and consists of four phases and each phase is represented by one quadrant. The four phases are planning, risk analysis, engineering and evaluation. During planning phase requirements are gathered. The objective of planning phase is to determine resources for the project as well as what functionality we want to incorporate in

ISSN:2319-6890)(online),2347-5013(print)
01 April 2015

the product. Cost and scheduling of product is also considered in it. In the risk analysis phase focus in on identification of risks and at the same time providing alternate solutions for them. Software is developed in the engineering phase and during evaluation it undergoes testing. The radius of the spiral represents the cost and angular dimension represents the progress in process. In this model aim is to identify high risks related to project and resolve it before it threatens the software operation or cost.

Risks related to over budget, delay, unmanageable and insufficient product quality must be taken into account and resolved before processing to next phase.

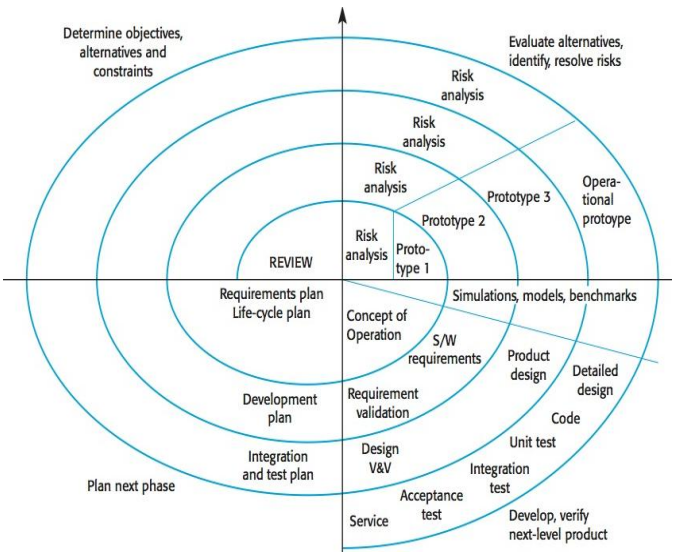

Figure6: Spiral Model

3.6 Build and Fix Model

In this model a software product is built without any specification and without applying any kind of design. Developer in this model adopts an adhoc**[4]** approach which is not well defined. Developer built the product as many times as possible until it satisfies the client. Build and Fix are two phases of the model. Build phase deals with writing the code and Fix phase deals with correcting it in correspondence to user requirements (functionality).

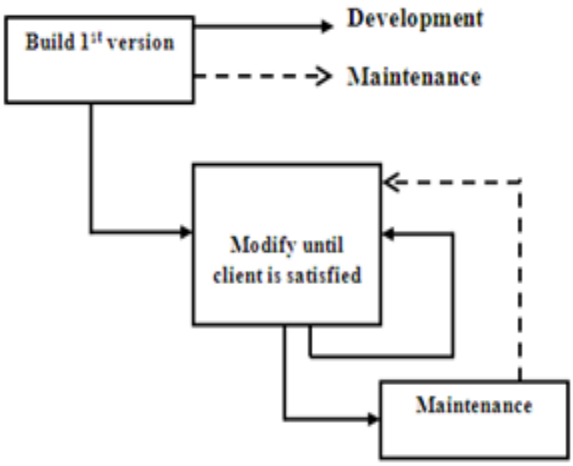

Figure7: Build and Fix Model

This model includes the two phases.

- **Build:** In this phase, the software code is developed and passed on to the next phase for the launch.
- **Fix:** In this phase, the code which has been developed in build phase is made error free. Also, in addition to the corrections to the code, the code is modified according to the user's requirements and recommendations.

Build and Fix model requires less experience to execute or manage other than the ability to program. It is suitable for smaller software. It requires less project planning.

3.7 V-Shaped Model

This model can be considered as extension of waterfall model. In waterfall model we move in a linear way while in V model process steps are bent upwards the coding phases to form typical V-Shape**[7]**. The relationship between each phase of the development process (verification) and its associated phase of testing (validation) is shown in V model. In V model emphasis is more on testing as testing is one of the important part of Software development.

V- Model application is almost same as waterfall model as both the models are sequential. Requirements have to be very clear before work on the project starts because it is usually expensive to go back and make changes for the existing projects. This model is used in the medical development field, as it is strictly disciplined domain. Following are the suitable scenarios where V-Model can be used:

- 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.

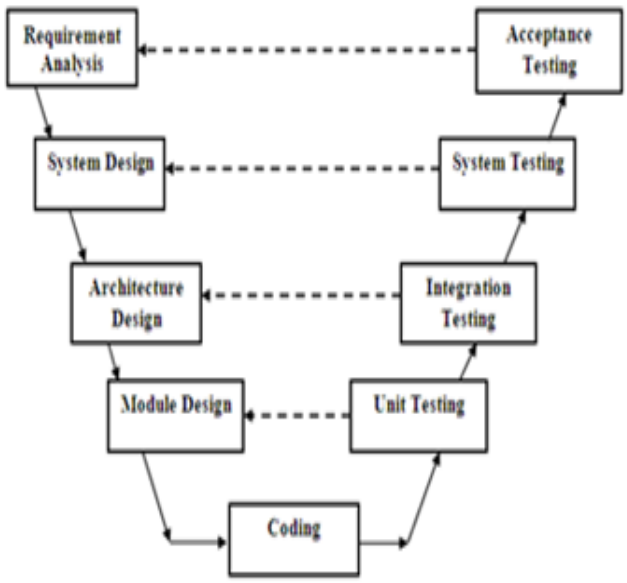

Figure8: V-Shaped Model

ISSN:2319-6890)(online),2347-5013(print)
01 April 2015

1V. CONCLUSION

There are many types of software development models such as Waterfall, Prototype, Spiral, V-Shaped model etc. Software development team will have to decide which model is to be used for their project as per customer requirements .All these models have there advantages and disadvantages. An organization which needs to develop a project in a linear fashion will use waterfall model. The important thing which organizations must take into consideration is that they should develop a 'bug' free product before launching it in market. Fusion of more than one model may be required in some software development product.

V. REFERENCES

i. Boehm, B. W. "A Spiral Model of Software Development and Enhancement", ISSN: 0018-9162, Volume: 21, Issue: 5, on page(s): 61-72, May 1988.

ii. Youssef Bassil, "A Simulation Model for the Waterfall Software Development Life cycle", International Journal of Engineering &Technology, ISSN: 2049-3444, Vol.2, No.5, 2012.

iii. Barry Boehm, "Spiral Development: Experience, Principles, and Refinements", edited by Wilfred J.Hansen, 2000.

iv. 15A. M. Davis, H. Bersoff, E. R. Comer, "A Strategy for Comparing Alternative Software Development Life Cycle Models", Journal IEEE Transactions on Software Engineering ,Vol. 14, Issue 10, 1988.

v. Roger S.Pressman,"Software engineering: A practitioner approach",ISBN 0-07-365578-3,5th ed., TMH, 2001.

vi. Ian Somerville, Software Engineering ,Addison Wesley,9th ed,2010.

vii. Nabil Mohammed Ali Munassar And A.Govardhan, "A Comparisonbetween Five Models of Software Engineering" Ijcsi International Journal of Computer Science Issues, Vol7, Issue 5, Sep, 2010.

viii. Craig Layman and Victor Basili, "Iterative and Incremental Development: A Brief History", IEEE Computer, 2003.

ix. W. Royce, "Managing the Development of Large Software Systems," presented at the Proceedings of IEEE WESCON, 1970.

IV. COMPARISON OF SDLC MODELS

Table I: Comparison of various SDLC Models on different Parameters

Model/ Features	**Waterfall**	**Prototype**	**RAD**	**Incremental**	**Spiral**	**Build &Fix**	**V-model**
Well defined requirements	Yes	No	Yes	No	No	No	Yes
User involvement in all phases	Only at beginning	High	Only at beginning	Yes(Intermediate)	High	No	No
Risk analysis	Only at beginning	No Risk analysis	Low	No Risk analysis	Yes	No	Only at beginning
Overlapping phases	No over-lapping	Yes	No	No	Yes	Yes	No
ImplementationTime	Long	Quick	Quick	Long	Long	Depend upon project	Long
Cost	Low	High	Low	Low	Expensive	Low	Expensive
Incorporation of changes	Difficult	Easy	Easy	Easy	Easy	Difficult	Difficult
Simplicity	Simple	Simple	Simple	Intermediate	Intermediate	Simple	Intermediate
Flexibility	Rigid	Little Flexible	High	Less Flexible	Flexible	Flexible	Less flexible