SOFTWARE DEVELOPMENT LIFECYCLE MODELS AND COMPARISON OF THAT MODELS

YAVRAJDEEP KAUR, DEVINDER KAUR

Abstract: This paper deals with a vital and important issue in software engineering world. It is concerned with the software development processes which are known as software development life cycle (SDLC). Software Development Life cycle models are very important for developing any system effectively and timely. Any software industry use different models according to their working environment and requirement. This paper indicates four software development life cycle models that is Waterfall model, spiral model; Iterative model and Prototype model .Every model has its own features, advantages and limitations. Comparison of these four models is also characterized in this paper.

Keywords: Comparison, SDLC, SDLC models, Software engineering.

Introduction: System Development Life Cycle (SDLC) is the most well-known of the so-called phase models. Still in use today, although rarely found in pure form. Process divided into five to eight distinctive phases. Each phase has as output documentation/programs. The systems development life cycle (SDLC), also referred to as the application development life-cycle. Each SDLC has its strengths and weaknesses, and each SDLC may provide better functionalities in one situation than in another. SDLC models vary greatly in their scope, end-user involvement, risk assessment, and quality control [4].
Software Engineering (SE) is the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software and the study of these approaches; that is, the application of engineering to software because it integrates significant mathematics, computer science and practices whose origins are in Engineering [1].Software Engineering is a discipline whose aim is the production of quality software, software that is delivered on time, within budget and that satisfies its requirements. Software Engineering is the area which is constantly growing. It is very interesting subject to learn as all the software development industry based on this specified area [2]. A software development process is a structure imposed on the development of a software product. There are several models for such processes, each describing approaches 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.
Software Engineering processes are composed of many activities, notably the following:

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

2. PHASES OF SDLC: Problem solving in software consists of these activities:

- Understanding the problem
- Deciding a plan for a solution
- Coding the planned solution
- Testing the actual program

For large systems, each activity can be extremely complex and methodologies and procedures are needed to perform it efficiently and correctly. Furthermore, each of the basic activities itself may be so large that it cannot be handled in single step and must be broken into smaller steps. For example, design of a large software system is always broken into multiple, distinct design phases. The basic activities or phases to be performed for developing software system are:

- Determination of System's Requirements
- Design of system
- Development (coding) of software
- System Testing [6]

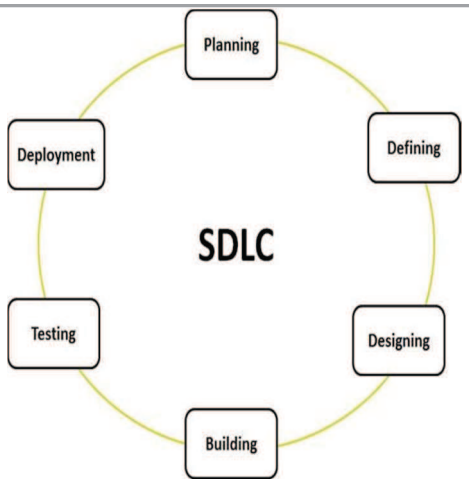

Fig.1 Phases of an SDLC model

3. Software Development Models:

A) Original Waterfall Model: Waterfall model was proposed by Royce in 1970 which is a linear sequential software development life cycle (SDLC) model. It seems to be the first named software development lifecycle model to be described and used. In this model whole work is done in linear fashion. Entire work is divided into five different phases. All the phases are cascaded to each other so that second phase is started as and when defined set of goals are achieved for the first phase and it is signed off, so it is named as "Waterfall Model". Requirements are very well understood before start working. It defines some basic tasks, which are carried out in sequence: requirements definition, architecture design, detailed design, implementation, component verification, integration verification and requirements validation. See Fig. 2

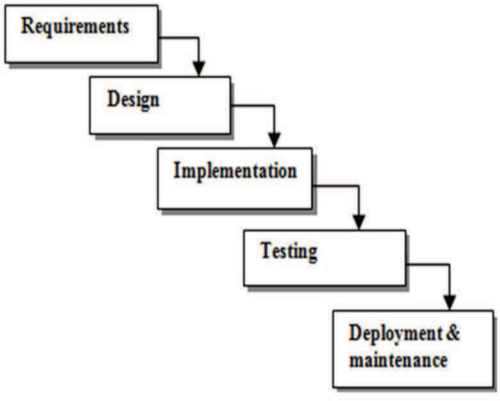

Fig. 2 Original Waterfall Model [3]

Advantage of original water model

- Each stage has well defined deliverable or milestone.
- It is simple to use and understand.

Disadvantage of waterfall model

Since the waterfall model has been around for over 30 years, it has been tested, to the large extent, and number of problems has surfaced. Critics complain that the waterfall model is:

- Unstable: since client requirements have to be fixed before the development process begins any requirement changes destabilizes the whole development process.
- Impossible to view the model: user can only view the system after implementation.
- Cheaper: since amount of resource required is very minimal.
- The biggest disadvantage of the waterfall model is one of its greatest advantages. You cannot go back a step; if the design phase has gone wrong, things can get very complicated in the implementation phase [1]-[8].

B) Iterative Waterfall Model: The problems with the Waterfall Model created a demand for a new method of developing systems which could provide faster results, require less up-front information and offer greater flexibility. Iterative model, the project is divided into small parts. This allows the development team to demonstrate results earlier on in the process and obtain valuable feedback from system users. Each iteration is actually a mini-Waterfall process with the feedback from one phase providing vital information for the design of the next phase [1].

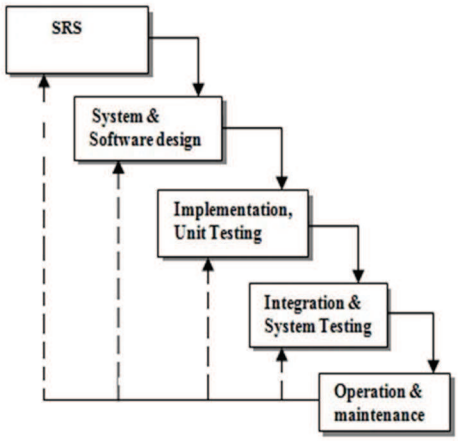

Fig. 3 Iterative Waterfall model

Advantages

- It allows feedback to proceeding stages.
- It can be used for project wherein the requirements are not well understood.

Disadvantages

- It is not easy to manage this model.
- No clear milestones in the development process.
- No stage is really finished [1].

C) Spiral Model

The spiral model was defined by Barry Boehm in his 1988 article A Spiral Model of Software Development

and Enhancement. This model was not the first model to discuss iterative development, but it was the first model to explain why the iteration matters. As originally envisioned, the iterations were typically 6 months to 2 years long. Each phase starts with a design goal and ends with the client (who may be internal) reviewing the progress thus far. Analysis and engineering efforts are applied at each phase of the project, with an eye toward the end goal of the project. This is the most advantageous and practical model of all the SDLC models. The whole model is an iterative spiral steps that is continuously repeated over and over time to generate the actual software components with each spiral. Thus help in reducing the complexity of the software being developed [4].

The spiral model has four phases: Planning, Risk Analysis, Engineering and Evaluation. A software project repeatedly passes through these phases in iterations. The baseline spiral, starting in the planning phase, requirements is gathered and risk is assessed. Each subsequent spiral builds on the baseline spiral. Requirements are gathered during the planning phase. In the risk analysis phase, a process is undertaken to identify risk and alternate solutions. A prototype is produced at the end of the risk analysis phase. Software is produced in the engineering phase, along with testing at the end of the phase. The evaluation phase allows the customer to evaluate the output of the project to date before the project continues to the next spiral. In the spiral model, the angular component represents progress, and the radius of the spiral represents cost [7].

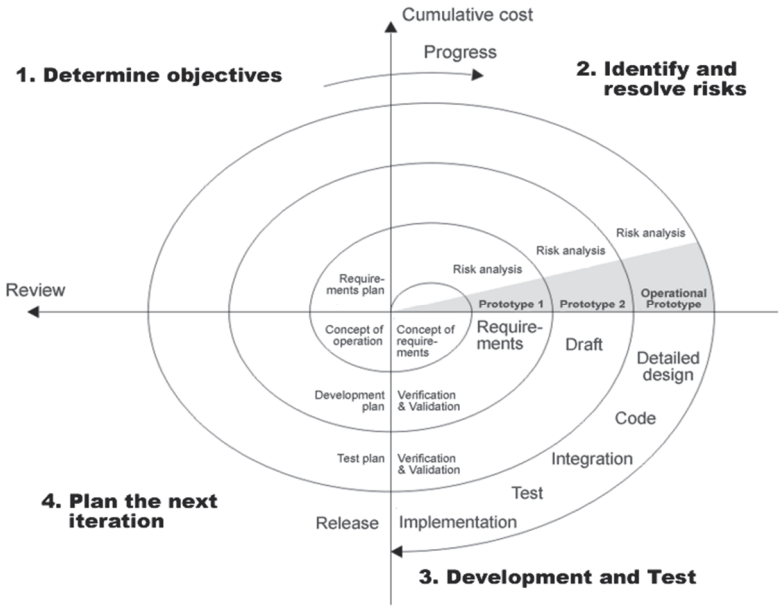

Fig. 4 Spiral Model [7]

Advantages

- Every time a new prototype is obtained, it is revaluated again and again by the customer .so more customer involvement is there.
- Better productivity through reuse capabilities.
- Proper control over cost, time and manpower requirement for a project work.
- Errors are eliminated in early phases of project development only.

Disadvantages

- This model requires risk identification, its projection, risk assessment and risk management which is not an easy task.
- Cost and time estimations are also not very easy.
- This model is not suitable for smaller project as then the cost of risk analysis is greater than cost of the entire project [1].

D) Prototyping Model: Software prototyping is the development approach of activities during software development, the creation of prototypes, i.e., incomplete versions of the software program being developed [1].

Prototype Model places more effort in creating the actual software instead of concentrating on documentation. This way, the actual software could be released in advance. Prototyping requires more user involvement and allows them to see and interact with a prototype allowing them to provide better and more complete feedback and specifications. The presence of the prototype being examined by the user prevents many misunderstandings that occur when each side believe the other understands what they said. The

final product is more likely to satisfy the user's desire for look, feel and performance. Incremental model is at the heart of a cyclic software development process. It starts with an initial planning and ends with deployment with the cyclic interactions in between. Easier to test and debug during a smaller iteration. Easier to manage risk because risky pieces are identified and handled during its iteration. Spiral model is good for large and mission critical projects where high amount of risk analysis is required like launching of satellite [5].

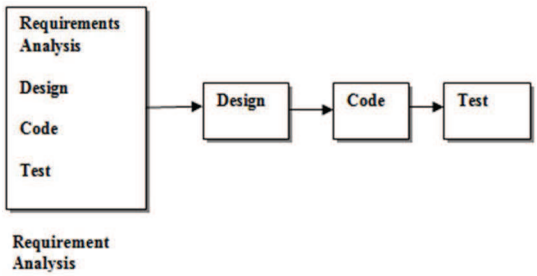

Fig. 5 Prototype Model

Advantages

- Users are actively involved in the development
- When prototype Model is shown to the user, he gets a proper clarity about his requirements. And feel the functionality of the software, so can suggest the changes and modifications.
- It reduces risk of failure, as potential risks can be identified early and steps can be taken to remove that risk.
- The customer does not need to wait long for working software.

Disadvantages

- Wastage of Time and money to build prototype, if client not satisfied.
- Too many changes can disturb the rhythm of the developer team.
- Long term procedure.
- It follows the "Quick and dirty" approach- the prototype is through away after showing to the client [2].

4. Comparison Of Different Sdlc Models: As there are various models of software development life cycle, each has its own advantages and disadvantages depending upon which we have to decide, which model we should choose. For instance if the requirements are known before hand and well understood and we want full control over the project at all time, then we can use waterfall model. Spiral model is good for large and mission critical projects where high amount of risk analysis is required like launching of satellite [2]. The comparison of the different models is represented following on the basis of certain feature.

Understanding Requirements: In Waterfall model and Spiral Model, developing organization is familiar with the problem domain and the requirements are very well understand, whereas in Iterative water fall and Prototyping model requirements are not known or change with time. So in those models requirements are known only with time.

Cost: Cost of Waterfall model is low as compared to Iterative model, Prototype model and Spiral model.

Availability of reusable component: Waterfall model has not availability of reusable components. Prototype model, Iterative waterfall model and Spiral model have Availability of reusable component.

Complexity of the system: Original waterfall model and Iterative waterfall model are simple and easy to understand. Where as Prototype and Spiral models are not simple because requirements has changed with time.

Risk Analysis: The Waterfall model has risk analysis only at beginning. Iterative waterfall and Prototype model reduces risk. Whereas in Spiral model risk analysis is high.

User Involvement in all phases of SDLC: The Waterfall model has user Involvement at initial phases. User Involvement in iterative waterfall has at intermediate phase. Prototype model and Spiral model have greater user involvement.

Guarantee of Success: The Iterative water model has high guarantee of success as compared to Original Waterfall model. Prototype model and Spiral model are successful models.

Implementation time: Cycle time for Waterfall model is too long. Iterative model and prototype models have less implementation time. Whereas implementation time for Spiral model is depend upon project.

Flexibility: Waterfall model is not flexible since requirements are known at early stage. Iterative waterfall model is less flexible. Prototype model and Spiral model are flexible.

Changes Incorporated: Original Waterfall model disallows changes. Prototype model disallows later changes. Whereas Iterative waterfall model and Spiral model are allows changes easily.

Conclusion: After study of all models through the various factors, it has been found that the original water fall model is used by various big companies for their internal projects. Since the development team is familiar to the environment and it is feasible to specify all requirements of working environment. Iterative water fall model overcome the drawback of original waterfall model. It allows feedback to proceeding stage. Prototype model used to develop online systems for transaction processing. Spiral model is used for development of large, complicated

and expensive projects like scientific projects .All these different software development models have their own advantages and disadvantages. The study of these models make easy to select the SDLC model for software industry.

References:

1. Ms. Shikha maheshwari , Prof.Dinesh Ch. Jain, "A Comparative Analysis of Different types of Models in Software Development Life Cycle," International Journal of Advanced Research in Computer Science and Software Engineering , Volume 2, Issue 5, May 2012.
2. Naresh Kumar, A. S. Zadgaonkar, Abhinav Shukla, "Evolving a New Software Development Life Cycle Model ," SDLC-2013 with Client Satisfaction International Journal of Soft Computing and Engineering (IJSCE) ISSN: 2231-2307, Volume-3, Issue-1, March 2013
3. Apoorva Mishra, Deepty Dubey, "A Comparative Study of Different Software Development Life Cycle Models in Different Scenarios," Volume 1, Issue 5, October 2013 International Journal of Advance Research in Computer Science and Management Studies.
4. Vishwas Massey ,Prof. K.J.Satao, "Comparing Various SDLC Models And The New Proposed Model On The Basis Of Available Methodology," International Journal of Advanced Research in Computer Science and Software Engineering.
5. James E. Purcell, CISSP, GSEC, GCIH, PMP, MCSE, "Comparison of Software Development Lifecycle Methodologies," This paper is from the SANS Software Security site.
6. Rajendra Ganpatrao Sabale, Dr. A.R. Dani ,"Comparative Study of Prototype Model For Software Engineering With System Development Life Cycle," IOSR Journal of Engineering (IOSRJEN) ISSN: 2250-3021 Volume 2, Issue 7(July 2012), PP 21-24
7. Gourav Khurana , Sachin Gupta "STUDY & COMPARISON OF SOFTWARE DEVELOPMENT LIFE CYCLE MODELS," IJREAS Volume 2, Issue 2 (February 2012) ISSN: 2249-3905.
8. Unnati A. Patel , Niky K. Jain, "New Idea In Waterfall Model For Real Time Software Development," International Journal of Engineering Research & Technology (IJERT) Vol. 2 Issue 4, April – 2013 ISSN: 2278-0181.

* * *

Student, Student,Pati Budh Singh Wala Village Rampura, distt. Bathinda, Punjab/ Baba Farid Group of Institution, Bathinda, Punjab, India, ssskaur10@gamil.com
devinderbrar90@gmail.com