Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 2, Issue. 1, January 2013, pg.33 – 38

SURVEY ARTICLE

SDLC Models- A Survey

Sukhdev Singh Ghuman
SBDSM Khalsa College Domeli (Kpt.)
ghumanggg@gmail.com

Abstract— Software Development Life Cycle (SDLC) is a model which provides us the basic information about the methods/techniques to develop software. It is concerned with the software management processes that examine the area of software development through the development models, which are known as software development life cycle. There are many development models namely Waterfall model, Iterative model, V-shaped model, Spiral model, Extreme programming, Iterative and Incremental Method, Rapid prototyping model and Big Bang Model. This is paper is concerned with the study of these different software development models and to compare their advantages and disadvantages.

Keywords: SDLC, Software, design, model, Programming

I. INTRODUCTION

Software development is one of the most difficult and time consuming task. It also requires a lot of labour. A computer is useless without the software, so software development activity require concerted effort and team work of a group of people. To guide the people about developing software, we use one of the many models available to us. Software is a set of programs for the purpose of facilitating works of offices, administrations, banks, etc. The aim of software engineering is to construct programs of high quality by using appropriate software process model. There are many different models which are discussed in this paper.

II. SOFTWARE PROCESS MODELS

A Software Development Life Cycle (SDLC) is a set of steps, along with ordering constraints on execution to produce software of desired quality. Many people are required for development of the system and to perform different activities for the completion of the project. The different models discussed in the paper are as given below:-

- **Waterfall model.**
- **Iterative model.**
- **V-shaped model.**
- **Spiral model.**
- **Extreme Programming**
- **Evolutionary Model**
- **Prototyping Model**
- **Incremental Method**
- **Rapid prototyping model**
- **Big Bang Model.**

A. The Waterfall Model

The waterfall model [1] is the classical model of software engineering. This model is one of the oldest models and is widely used for software development. As this model emphasizes planning in early stages, it ensures design flaws before they develop. In addition, its intensive document and planning make it work well for projects in which quality control is a major concern**.** The model pass through the following phases feasibility study, requirements analysis, design, detailed design, coding, testing, and maintenance.

Advantages: Simple easy to use and understand. It reinforces good habits such as define before- design, design-before-code. It identifies deliverables and milestones, Document driven, Published documentation standards, Works well on smaller software projects.

Disadvantages: It is inflexible, slow and costly. It is unrealistic to expect accurate requirements so early in project. Software is delivered late in project, delays discovery of serious errors. It is difficult to integrate risk management. It is expensive to make changes to documents.

B. Iterative Model

An iterative life cycle model [3] does not attempt to start with a full specification of requirements. Instead, development begins by specifying and implementing just part of the software, which can then be reviewed in order to identify further requirements. This process is then repeated, producing a new version of the software for each cycle of the model.

Advantages: In iterative model build and improve the product step by step, we can track the defects at early stages. This avoids the downward flow of the defects. In iterative

model we can get the reliable user feedback. In iterative model less time is spent on documenting and more time is given for designing.

Disadvantages: Each phase of iteration is rigid with no overlaps, costly system architecture or design issues may arise because not all requirements are gathered up front for the entire life cycle.

C. V-Shaped Model

It is called V shaped model it represents V shape. Like waterfall model, the V-Shaped life cycle [2] is a sequential path of execution of processes. Each phase must be completed before the next phase begins. Testing is important in this model than in the waterfall model. The testing procedures are developed early in the life cycle before any coding is done, during each of the phases preceding implementation. Requirements begin the life cycle model just like the waterfall model. Before development is started, a system test plan is created. The test plan focuses on meeting the functionality specified in requirements gathering. The high-level design phase focuses on system architecture and design. An integration test plan is created in this phase in order to test the pieces of the software systems ability to work together.

Advantages: Simple and easy to use. Each phase has specific deliverables. Higher chance of success over the waterfall model due to the early development of test plans during the life cycle. Works well for small projects where requirements are easily understood.

Disadvantages: Very rigid like the waterfall model. Little flexibility and adjusting scope is difficult and expensive. Software is developed during the implementation phase, so no early prototypes of the software are produced. This model does not provide a clear path for problems.

D. Spiral model

The spiral model [4] is similar to the incremental model, with more emphasis placed on risk analysis. The spiral model has six phases: Customer Communication, Planning, Risk Analysis, Engineering, Construction and release and Customer Evaluation. A software project repeatedly passes through these phases in iterations (called Spirals in this model). The baseline spiral, starting in the planning phase, requirement 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 data 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.

Advantages:
It estimates of budget and schedule is more realistic. It is more able to cope with changes. It helps in risk avoidance. Good for large and mission-critical projects. Software is produced early in the software life cycle.
Disadvantages: Can be a costly model to use. Risk analysis requires highly specific expertise. Project's success is highly dependent on the risk analysis phase. Not suitable for smaller projects.

E. Extreme Programming.
An approach to development [5] based on the development and delivery of very small increments of functionality. It relies on constant code improvement, user involvement in the development team and pair wise programming. It can be difficult to keep the interest of customers who are involved in the process. Team members may be unsuited to the intense involvement that characterizes agile methods. Prioritizing changes can be difficult where there are multiple stakeholders. Maintaining simplicity requires extra work. Contracts may be a problem as with other approaches to iterative development.

Advantages: Lightweight methods suit small medium size projects. Produces good team cohesion and emphasizes final product and Iterative. Test based approach to requirements and quality assurance.

Disadvantages: Difficult to scale up to large projects where documentation is essential and needs experience and skill if not to degenerate into code-and-fix. Programming pairs is costly.

F. Incremental Model
It is developed to overcome the weaknesses of the waterfall model. It starts with an initial planning and ends with deployment with the cyclic interactions in between. The basic idea behind this method is to develop a system through repeated cycles (iterative) and in smaller portions at a time (incremental), allowing software developers to take advantage of what was learned during development of earlier parts or versions of the system [6].

Advantages: Produces business value early in the development life cycle· Better use of scarce resources through proper increment definition. Can accommodate some change requests between increments. More focused on customer value than the linear approaches. Problems can be detected earlier.

Disadvantages:
It requires heavy documentation and follows a defined set of procedures, defines increments based on function and feature dependencies. It requires more customer involvement than the linear approaches. Partitioning the functions and features might be problematic. Integration between iteration can be an issue if this is not considered during the development.

G. Evolutionary Model

It is called prototyping or simulation model. It shows software development as a series of rises, each representing a separate loop of spiral. It shows that loops or releases tend to overlap each other that make it clear that development work tends to reach a peak, at around the time of the deadline for completion.

Advantage:
It is simple to understand. Early delivery of the parts of the system is possible.

Disadvantages:
It takes more time as compared to other lifecycle models. It is very difficult to run parallel and it takes a lot of effort to implement.

H. Rapid prototyping model or Rapid Application Development (RAD)

A rapid prototype [7] is a working model that is functionally equivalent to a subset of the product. Because the working prototype has been validated through interaction with the client, the resulting specification will be correct. Verification is needed in specification, planning, and design. In implementation and integration, testing is needed. An essential aspect of a rapid prototype is in the word *rapid*. We can combine waterfall and rapid prototyping, by using rapid prototyping to find out the client's requirements.

Advantages: RAD reduces the development time and reusability of components help to speed up development. All functions are modularized so it is easy to work with.

Disadvantages: For large projects RAD require highly skilled engineers in the team. Both ends Customer and developer should be committed to complete the system in a much abbreviated time frame. RAD is based on Object oriented approach and if it is difficult to modularize the project the RAD may not work well.

I. Prototyping Model:

Prototype model is started by developing a small prototype and followed by a mini waterfall process, primarily together requirements. The first prototype is reviewed in subsequent loops. The project team performs further requirements, design, implementation and review.

Advantages: The system is delivered quickly and users are involved in the whole system development process.

Disadvantage: Prototype is very difficult to control and manage. The final system also creates problem during maintenance.

J. The Big Bang Model

The Big- Bang Model [8] is just like cosmological model in which the one in which huge amount of people or money is put together, a lot of energy is expended and out comes the perfect software product or it doesn't. The beauty of this model is that it's simple. There is little planning, scheduling, or formal development process. All the effort is spent developing the software and writing the code. It is an ideal process if the product requirements aren't well understood and the final release date is flexible. It is also important to have flexible customers, too, because they won't know what they're getting until the very end.

III. CONCLUSION

In this paper various software development life cycle models are studied and compared. The Waterfall model provides base for other development models. The various models such as Iteration model, V-shaped model, Spiral model, Extreme programming, Evolutionary Prototyping Model, Iterative and Incremental Method, Rapid prototyping model, The Big Bang Model, and code and fix models are compared.

REFERENCES

[1] Royce, Winston, "Managing the Development of Large Software Systems", Proceedings of IEEE WESCON 26, 1970.

[2] Kevin Forsberg and Harold Mooz, "The Relationship of System Engineering to the Project Cycle," in Proceedings of the First Annual Symposium of National Council on System Engineering, October 1991: 57–65.

[3] Craig Larman and Victor R. Basili,"Iterative and Incremental Development: A Brief History". IEEE Computer (IEEE Computer Society) 36 (6): 47–56. Doi:10.1109/MC.2003.1204375. ISSN 0018-9162

[4] Boehm B, "A Spiral Model of Software Development and Enhancement", ACM SIGSOFT Software Engineering Notes", "ACM", 11(4):14-24, August 1986.

[5] Mohamed Sami Abd EI-Satar" Software Development Life Cycle Models and Methodologies", 2012, http://melsatar.wordpress.com

[6] Craig Larman and Victor Basili, Iterative and Incremental Development: A Brief History, IEEE Computer, June 2003.

[7] C. Melissa McClendon, Larry Ragout, Gerri Akers: The Analysis and Prototyping of Effective Graphical User Interfaces. October 1996.

[8] Wollack, Edward J. "Cosmology: The Study of the Universe". Universe 101: Big Bang Theory. NASA, 2010

