JCSE International Journal of Computer Sciences and Engineering Open Access

Research Paper Vol.-7, Issue-9, Sept 2019 E-ISSN: 2347-2693

A Comparative Analysis of Software Development Models Based On Various Parameters

Archana Srivastava

Department of Computer Applications, BBD University, Lucknow, India

**Corresponding Author: sri_archee@yahoo.com*

DOI: https://doi.org/10.26438/ijcse/v7i9.1721 | Available online at: www.ijcseonline.org

Accepted: 12/Sept/2019, Published: 30/Sept/2019

Abstract— Gradually I.T industry is increasingly relying on a growing quantity of ever-larger software. The fashion of growing technical complexity of the systems united with the requirement for repeatable and predictable process methodologies have driven system developers to establish system development models. With the rising operations of organizations, the need to automate the various activities increased. With this need the concept of system lifecycle models came into survival that emphasized on the need to follow some structured approach towards building new or improved system. *At present the software systems cannot be built with mathematical or physical certainty so* all software systems are imperfect because they cannot be built with mathematical or physical certainty. *The set of processes those proved to be effective and efficient for software development in one organization may or may not be followed in another organization. That is other organization finds another approach for software development more convenient to work with.* The software development process is the very complicated thing without any proper step by step generating procedure so to make the software development processes simple and systematic the software development life cycle came in to existence. This is the systematic and structural method of software developing process. The SDLC defines the framework that includes different activities and tasks to be carried out during the software development process. *The development lifecycle of software Comprises of four major stages namely Requirement Elicitation, Designing, Coding and Testing. This workflow is a guideline for successful planning, organization and final execution of the software project.*
There are various software development life cycle models that are used in the software development process heaving their own advantages and disadvantages.

Keywords: Software Development Life Cycle (SDLC), Models and Comparative Analysis, Activities involved in SDLC models, Model with Different parameter, factors affecting to Chose Process Model, SRS (software requirement specification), SRD (System requirement documentation), Software Process, Risk Analysis, Verification and Validation.

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 **[1]** 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

Software Engineering Terminology in IEEE standard Glossary, the software lifecycle is "The period of time that starts when a software product is conceived and ends when the product is no longer available for use".

According to Elliott (2004) the systems Process life cycle (SDLC) can be considered to be the oldest formalized model framework for building information systems. The main idea of the SDLC has been to pursue the process of information systems in a very deliberate, structured and methodical way [2]. The main target of this model framework in the 1960s was to develop large scale functional business systems.
There are five major era of Software Process Model:
1. Mainframe Era[1960-1970]
2. Midrange Era[1970-1980]
3. Microcomputer Era[1980-1990]
4. Internet Era[1990-2000]
5. Personalized Era[2000-Till Now]

Software Engineering (SE) is the application of a organized, closely controlled, quantifiable approach to the development, operation, and maintenance of software, and the study of these approaches; means, the application of engineering to software because it integrate important mathematics, computer science and practices whose origins are in Engineering. A methodology is a organized way of doing things. It is a repetitive process that we can pursue from the earliest stages of software development through to the last stage i.e. maintenance of an installed system. It is broadly accepted that no single approach that will prevent project overruns and failures in all cases.

IT product development has its own challenges – hurdles because it required an entirely different approach. Different approach is required because of its inherent quality of being intangible in nature [10].
The software development cycle is all about.

 Problem domain i.e. Understanding the problem.
 Solution domain i.e. Decide a plan for solution.
 Designing and Coding of planned solution.
 Test the actual program.
 Maintain the product.

Rest of the paper organization as follows: Section I contains the introduction of Software Engineering, Section II contains the related work of Comparison between different models, Section III contains literature Survey about different models, Section IV contains Comparative analysis between various models, Section V concludes research work with future directions

II. RELATED WORK

Various Software models are studied in different research papers and concluded on different parameters. Few models are found with drawbacks and few are with some advantages.

III. DIFFERENT SDLC MODELS

WATERFALL MODEL

Waterfall approach was first SDLC Model to be used broadly in Software Engineering to make sure the success of the project. In "The Waterfall" approach, the entire process of software development is divided into separate phases. In the Waterfall model, in general, the result of one phase behaves as the input for the next phase sequentially.
The sequential phases in Waterfall model are −

- **Requirement Gathering and analysis**.
- **System Design**
- **Implementation**
- **Integration and Testing**
- **Deployment of system**
- **Maintenance**.

Application
Every software developed is different and require a appropriate SDLC approach to be followed based on the internal and external factors. Following are few situations where the use of Waterfall model is most appropriate are −

- Requirements are quite well documented, obvious and fixed.
- Product definition is stable.
- Technology is understood and is not dynamic.
- There are no ambiguous requirements.
- Plenty resources with required expertise are available to support the product.
- The project is short.

ITERATIVE MODEL

In the Iterative model, iterative process starts with a simple implementation of a little set of the software requirements and iteratively enhances the growing versions until the entire system is implemented and ready to be deployed.

An iterative life cycle model does not try to start with a complete specification of requirements. Instead, development starts by specifying and implementing just part of the software, which is then reviewed to recognize further requirements. This process is then repeated, producing a new version of the software at the end of each iteration of the model.

Application
This model is most often used in the following scenarios −

- Requirements of the entire system are clearly defined and understood.
- Main requirements should be defined; however, some functionalities or requested enhancements may evolve with time.
- There is a time to the market restriction.
- A new technology is used and is being learnt from the development team during working on the project.
- Resources with required skill sets are not available and are planned to be used on contractual basis for explicit iterations.
- There are few high-risk features and goals which may change in the future.

SPIRAL MODEL

The spiral model have four phases. A software project repetitively passes through these phases in iterations called Spirals.

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

Application

The below mentioned points explain the typical uses of a Spiral Model −

- When there is a financial constraint and risk evaluation is important.
- For middle to high-risk projects.
- Long-term project promise because of possible changes to economic priorities as the requirements change with time.
- Customer is not certain of their requirement which is usually the case.
- Customer necessities are complex and need evaluation to get clarity.
- Fresh product line which should be released in phases to get enough customer feedback.
- Important changes are expected in the product during the development cycle.

V-MODEL

V-model is an SDLC model in which execution of processes happens in a sequential manner in a V-shape. It is also famous as Verification and Validation model.

The V-Model is an expansion of the waterfall model and is based on the association of a testing phase for each corresponding development stage. This means that for each single phase in the development cycle, there is a directly linked testing phase. This is a high-disciplined model and the next phase starts only after completion of the previous phase.

Application

The below mentioned points are some of the most suitable scenarios to use the V-Model application.

- Requirements are very clearly defined, clearly documented and fixed.
- Product definition is constant.
- Technology is not dynamic and is well unstated by the project team.
- There are no ambiguous or undefined requirements.
- The project is very short in length.

BIG BANG MODEL

The Big Bang model is an Software Development Life Cycle model in which we do not follow any specific process. The development just begins 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. This Big Bang Model does not follow a specific procedure and there is a very slight planning required. Even the customer is not sure about what accurately he wants and the requirements are implemented on the fly without much analysis.

AGILE MODEL

The Agile Software Development Life Cycle model is a combination of incremental and iterative process models with focus on process adaptableness and customer satisfaction by quick delivery of working software product. Agile Methods breaks the product into small incremental builds. These builds are provided in iterations. Each iteration normally lasts from about one to three weeks. Every iteration involves cross functional teams working concurrently on various areas like −

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

At the end of the iteration, a running product is displayed to the customer and important stakeholders.

RAD Model

The RAD (Rapid Application Development) model is based on prototyping and iterative development with no precise planning involved. The process of lettering the software itself involves the planning required for making the product.

Rapid Application Development emphasises on gather customer requirements by workshops or focus groups, early testing of the prototypes by the customer using iterative concept, reuse of the existing prototypes (components), continuous integration and rapid delivery.

Application

RAD model can be applied effectively to the projects in which clear modularization is possible. If the project cannot be divided into modules, RAD may fail.

The below mentioned pointes describe the typical scenarios where RAD can be used −

- RAD must be used only when a system can be modularized to be delivered in an incremental manner.
- It must be used if there is a high availability of designers for modeling.
- It must be used only if the budget permit use of automated code generating tools.
- RAD SDLC model must be chosen only if domain experts are available with related business knowledge.
- Should be used when the requirements change during the project development and working prototypes are to be presented to customer in small iterations of few months.

SOFTWARE PROTOTYPE MODEL

The Software Prototyping refers to developing software application prototypes which displays the functionality of the product under development, but may not actually hold the exact logic of the original software.

Software prototyping is becoming very popular as a software development model, as it enables to understand customer requirements at an early stage of development. It helps get valuable feedback from the customer and helps software designers and developers understand about what exactly is expected from the product under development.

Following is a stepwise approach explained to design a software prototype.

- Basic Requirement Identification
- Developing the initial Prototype
- Review of the Prototype
- Revise and Enhance the Prototype

IV. COMPARATIVE ANALYSIS

TABLE 1: COMPARISION OF VARIOUS MODELS

Feature/Model	**Waterfall**	**Iterative**	**Spiral**	**V-Model**	**Bing-Bang**	**Agile**	**RAD**	**S/W Prototype**
Simplicity	Simple	Simple	Medium	Medium	Simple	Complex to use	Simple	Low
Success Guarentee	Low	High	High	High	Good	Very High	Good	Good
Flexibility	No	Good	Yes	Low	Yes	High	Yes	No
Changes to Incorporate	Difficult	Easy	Easy	Difficult	Very Difficult	Difficult	Easy	Easy
Expertise Staff	High	High	High	Medium		Very High	High	Medium
Documentation & Training	Vital	Weak	Yes	Yes		Yes	Limited	Weak
Maintenance	Least Glamarous	Easily	Typical	Least	Easily	Promote Maintainability	Easily Maintainable	Promote Maintainability
Understandability	Well Understood	Well Understood	Well Understood	Easily Understood	Well Understood	Well Understood	Easily Understood	Not Well Understood
Security	Vital	Limited	High	Limited	Weak	Demonstrable	Vital	Weak
Development Time	Long	Medium	Long	Medium	Medium		Quick	Long
Expertise Required	High	High	High	Medium	Medium		High	Medium
System Complexity	Simple	Simple	Complex	Bit Complex	Complex	Complex	Highly Complex	Complex
Interface	Minimum	Critical	Critical	Minimum	Minimum	Model Driven	Minimum	Critical
Time to Develop System	Long	Medium	Long	Acc to Project Size	Quick	Least Possible	Quick	Long
Requirement Specification	At Early Stage	At Early Stage	At Early Stage	At Early Stage	Frequently changes	Frequently changes	Time Box Released	Frequently changes
Understanding Requirement	Well Understood	Not Well Understood		Easily Understood	Easily Understood	Well Understood	Easily Understood	Not Well Understood
Possibility of Requirement Modification	Not Possible	Possible	Not Possible	Possible	Possible	Possible	Possible	Possible
Cost	Low	Low	High	High		Very High	Low	High
Risk Analysis	Only at Begining	Low	Yes	Only at Begining		Yes	Low	No Risk Analysis
Reusability	Limited	Yes	Yes	To Some Extent		Use Case Reuse	To Some Extent	Yes
User Involvement	Only at Requirement Analysis	Yes	High	No	No	High	Yes	High

Overlapping Phases	No	Yes	Yes	No	No	Yes	No	Yes
Success Rate	Low	High	High	High	Low	Very High	Good	Good
Flexibility	Rigid	Highly Flexible	Flexible	Little Flexible	Flexible	Highly Flexible	Flexible	No
Cost Control	Yes	No	Yes	Yes	No	Yes	No	No

V. CONCLUSION AND FUTURE SCOPE

In this paper I have tried to compare various SDLC models based on various parameters. SDLC models are apparatus that allow the development team to correctly follow the SDLC steps to create software that meets a business need. Each SDLC model has evolved as a new technology and has addressed weaknesses of older models. There are various models used in various organizations according to user requirements plus according to performance of a software. In present scenario waterfall and spiral are the most commonly used in the software development process and the other models are used according to the requirements of the software products All these different software development models have their own merits and demerits. However, in the commercial software development world, the fusion of all these methodologies is integrated. Timing is very important in software development. If there is delay happens during the development phase, the market could be taken over by the competitor. Also if a 'bug' filled product is launched in a short period of time (quicker than the competitors), it may affect the reputation of the company. So, there must be a trade-off between the development time and the quality of the product. Customers expect the S/W that can have few bugs but it should be user friendly.

No such model if found satisfying all conditions. In future we can develop a model to remove all disadvantages so it would be a quality product.

REFERENCES

[1] Ian Somerville, "*Software Engineering*" ,Addison Wesley,9th ed,2010.

[2] Garg, P.K. and W. Scacchi, ISHYS(1989) "*Design of an Intelligent Software Hypertext Environment*", IEEE Expert, Japan, April 1989.

[3] Sushma Malik1, Charul Nigam2, "*A Comparative study of Different types of Models in Software Development Life Cycle*", IJRET, Volume: 04 Issue: 10

[4] "*A Comparison between Five Models Of Software Engineering*", IJCSI International Journal of Computer

[5] Ms. Shikha Maheshwari, Prof Dinesh Ch. Jain, "*A Comparative Analysis of Different types of Models in Software Development Life Cycle*", IJARCSSE, Volume 2, Issue 5, May 2012 ISSN: 2277 128X

[6] Asmita, Kamlesh, Usha, *"Review On Comparative Study Of Software Process Model"*, International Journal of Science, Technology & Management,Volume No 04, Special Issue No. 01, March 2015

[7] Mr. Ashish Kumar Gupta, "*A Comparison Between Different Types Of Software Development Life Cycle Models In Software Engineering"*, International Journal of Advanced Technology in Engineering and Science , Volume No 03, Special Issue No. 01, March 2015.

[8] K. K. Aggarwal, Yogesh Singh, "*Software Engineering*" 3rd Edition.

[9] Manzoor Ahmad Rather ,Mr. Vivek Bhatnagar, "*A Comparative Study Of Software Development Life Cycle Models*", International Journal of Application or Innovation in Engineering& Management , Volume 4, Issue 10, October 2015

[10] Sushma Malik ,Charul Nigam, "*A Comparative study of Different types of Models in Software Development Life Cycle*", International Research Journal of Engineering and Technology, Volume: 04 Issue: 10 | Oct -2017

[11] Manzoor Ahmad Rather, Mr. Vivek Bhatnagar, *"A Comparative Study Of Software Development Life Cycle Models"*, International Journal of Application or Innovationin Engineering & Management,Volume 4, Issue 10, October 2015

[12] Shubham Dwivedi, *"Software Development Life Cycle Models - A Comparative analysis"*, International Journal of Advanced Research in Computer and Communication Engineering Vol. 5, Issue 2, February 2016.

[13] Prateek Sharma1, Dhananjaya Singh, *"Comparative Study of Various SDLC Models on Different Parameters"*, International Journal of Engineering Research ,Volume No.4, Issue No.4

[14] Ratnmala R. Raval,Haresh M. Rathod, *"Comparative Study of Various Process Model in Software Development "*, International Journal of Computer Applications (0975 – 8887) Volume 82 – No 18, November 2013

[15] Harminder Pal Singh Dhami, *"Comparative Study and Analysis of Software Process Models on Various Merits"*, International Journal of Advanced Research in Computer Science and Software Engineering Research Paper , Volume 6, Issue 9, September 2016

[16] Roger Pressman, titled *"Software Engineering - a practitioner's approach".*

[17] Hari Krishnan Natarajan, Ram Kumar Somasundaram and Kalpana Lakshmi, "A Comparison between Present and Future Models of Software Engineering", IJCSI International Journal of Computer Science Issues, Vol.10, Issue.2, 2013. Manzoor

[18] Chetna Sisodiya, Pradeep Sharma, *"Scrutiny to Effectiveness of Various Software Process Model"*, International Journal of Scientific Research in Computer Science and Engineering, Vol.5, Issue.2, pp.88-93, April(2017).

[19] Aanchal, Sonu kumar, *"Metrics for Software Components in Object Oriented Environments: A Survey"*, International Journal of Scientific Research in Computer Science and Engineering, Volume-1, Issue-2, March-April-2013.

Authors Profile

Mrs. Archana Srivastava Completed Bachelor of Science in 1996 and Master of Computer Application in 2005. She had completed her M.Tech in 2010. Currently pursuing Ph.D, and currently working as Associate Professor, School of Computer Applications, Babu Banarsi Das University, Lucknow, India since 2005. Her main research work focuses on Software Engineering. She has more than 15 years of teaching experience.