Volume-6, Issue-1, January - 2017 • ISSN No 2277 - 8160 IF : 3.62 | IC Value 80.26

Original Research Paper Computer Science

Our-Approach: A Novel Software Development Life Cycle Model

Md Ashif Habibi	Assistant Professor, Department of CSE, WIT, LNMU, Darbhanga, Bihar, India
Deependra Kumar Jha	Assistant Professor, Department of CSE, WIT, LNMU, Darbhanga, Bihar, India
Gita Sinha	Assistant Professor, Department of CSE, WIT, LNMU, Darbhanga, Bihar, India
Amar Choudhary	Assistant Professor, Department of Electronics Engineering, WIT, LNMU, Darbhanga, Bihar, India

ABSTRACT *Software Development Life Cycle is used to describe the period of time that starts with the software system being conceptualized and ends with the software being discarded after usage. There are many life cycle models and each one has its own merits and demerits. Unlike other life cycle models the waterfall model follows step by step process. So, change in requirements is not possible when the requirement phase is finished. Meanwhile, all other models do not follow step by step process. In order to address these issues, this paper presents OUR-Approach methodology that aims to design software development life cycle which copes with and handle continuous changes in requirements, follows step by step process, and deploys the product in a single iteration.*

KEYWORDS : SDLC, Step by Step Process, Requirements, Single Iteration, Deploy, Software Development Process

1. INTRODUCTION

Software Development Life Cycle (SDLC) may be an abstract model employed in project management that describes the stages concerned in a data system development project, from an initial practice baleens study through maintenance of the finished application. Software system life cycle models organize varied tasks of software system engineering into phases [11] [12]. A framework that describes the activities performed at every stage of a software system development project. Software system life cycle models specify however these phases area unit to be dead as well as the way within which these phases and tasks might restate and overlap [14].

In existing software system life cycle models, there are units many problems that require being self-addressed, like to deal with and handle continuous changes in necessities and follow step by step method. As an example, water model follows Step by Step Process so, change in requirements is not possible whereas all other models (Prototype, RAD) do not follow step by step process.

2. SOFTWARE LIFE CYCLE

2.1 Software Development Life Cycle (SDLC)

Software Development Life Cycle (SDLC) methodology is a formalized, standardized, documented set of activities used to manage a system development project [4]. It is the process in which you encapsulate your software development [13]. A framework that describes the activities performed at each stage of a software development [14]. SDLC is a standardized format for planning, organizing, and running a new development project. Development speed (time to market), Product quality, Project visibility, Risk exposure depends upon on SDLC. Normally, a lifecycle model covers the entire lifetime of a product.

2.2 Life Cycle Issues

Waterfall: Linear framework type. The waterfall model is a sequential design process in which progress is seen as flowing steadily through phases of requirements, analysis, design, construction, testing, deploy and maintenance [11][13]. The model has no mechanism to handle changes to the requirements that are identified because of user feedback.

Incremental: Combination of linear and iterative framework type. In this model, phases out deliveries by increment. The first increment is the core product of the system. Each further increment modifies the product to provide further functionality and features in system. Incremental model combines the elements of the waterfall model with the iterative philosophy of prototyping [11] [14]. When utilizing a series of waterfalls for a small part of the system before moving onto the next increment, there is usually a lack of overall consideration of the business problem and technical requirements for the overall system.

V Model: Verification and Validation Phases. This model relates each development phase to its associated testing phase. In this model, work on the testing phases is carried out in parallel. All other features are same as waterfall [5].

Prototype: Iterative framework type. The prototype model allows the user to see the prototype of the system early [13]. The goal of prototyping approach is to develop a little or pilot version referred to as an example of half or all of a system. But incomplete or inadequate problem analysis, resulting in current inefficient practices being easily built into the new system; it do not follow step by step process, Increases complexity of the overall system and Cost expenses.

Rapid Application Development (RAD): Iterative framework type. This uses minimal planning in favor of rapid prototyping. Each application in a system is given to separate teams. But it is very difficult to achieve consistency within and between applications developed by the different teams [14].

Spiral: Combination of linear and iterative framework type. The Spiral model is similar to the incremental model, with a lot of emphases placed on risk analysis. The Spiral model has four phases: Planning, Risk Analysis, Engineering, and Evaluation [11]. Risk analysis requires highly specific expertise and project's success is highly dependent on the risk analysis. No established controls for moving from one cycle to another cycle. While not controls, every cycle might generate a lot of work for consecutive cycle. No firm

deadlines and cycles continue with no clear termination condition, so there is an inherent risk of not meeting budget or schedule.

Rational Unified Process (RUP): Iterative framework type, Unified Process. The team members need to be expert in their field to develop software. The development process is too complex and disorganized. Development can get out of control, cost expenses, critical risk in the early stages, success of the project is not guaranteed [11][13].

3. STEP BY STEP PROCESS

Regular Process flows steadily through phases of Requirements, Analysis, Design, Coding, Testing, Deploy, and Maintenance [2] [3]. Most Professional preferred way is to follow step by step processes for successful project deploy. But step by step processes do not cope with and handle continuous changes in requirements.

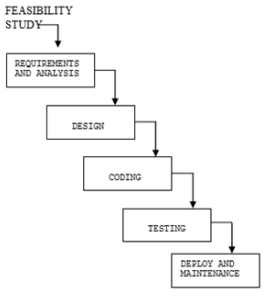

Figure 1. Step by Step Process

4. Our-Approach

Our-Approach methodology that aims to design SDLC which copes with the continuous changes in requirements, follows step by step processes, deploys the product in a single iteration and is possible to upgrade existing software product. The major difference between OUR-Approach and the Incremental model is that OUR-Approach completes tasks in a single iteration and the Incremental model completes tasks over a series of iterations to become the complete system.

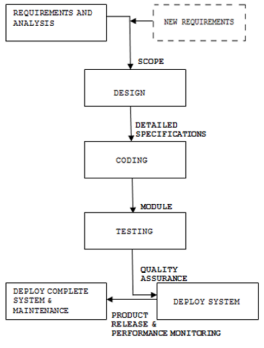

Figure 2. Our-Approach

Table 1

FEATURES	PROBABILITY
Guarantee of Success	Very High
UnderstandingRequirements	WellUnderstood
Cost	Low
RequirementSpecification	FrequentlyChanging
ResourceControl	No
CostControl	Yes
Iteration	Single
Simplicity	Simple
RiskInvolvement	EasilyManage
ExpertiseRequired	Medium
ChangesIncorporated	Easy
RiskAnalysis	Yes
UserInvolvement	InAllthePhases
OverlappingPhases	Yes
Flexibility	HighFlexible
Maintenance	PromotesMaintainability
Integrity	Easy
Reusability	Yes
TimeFrame	Short
Documentation&TrainingRequired	Yes

In Our-Approach, the time frame will be very short and so the product can be delivered in a shorter time. Success of the project is guaranteed with the advantage of step by step processes. Cost will be very low because of single iteration. It is very simple to implement. After every major stage, testing is done to check the correctness so as to prevent a bug/error. Timing is very crucial in software development; OUR-Approach will be tradeoffs between the development time and the quality of the product. It would be favored in projects where cost, schedule and quality are very important.

5. Our-Approach FEATURES

Our-Approach features such as handles continuous changes in requirements, iteration, understanding requirements, integrity, time frame and cost, and probability are illustrated in Table 1 [7] [8] [9] [10].

6. CONCLUSION

In this paper, I discuss a new way of SDLC: Our-Approach. Our-Approach will be tradeoffs between the development time and the quality of the product. Our-Approach would be favored in projects where iteration should be single and cope with and handle changes in requirements. It would be preferred in projects where Cost, Quality, Schedule, Success is very important.

REFERENCES

1. A. M. Davis, H. Bersoff, E. R. Comer, "A Strategy for Comparing Alternative Software Development Life Cycle Models", Published in IEEE Transactions on Software Engineering, 14(10):1453-1461, 1988
2. Horie, D.; Kasahara, T.; Goto, Y.; Jingde Cheng "A New Model of Software Life Cycle Processes for Consistent Design, Development, Management, and Maintenance of Secure Information Systems" Published in: International Conference on Computer and Information Science, 2009.
3. Software Process Models. Ian Sommerville. Published in: ACM Computing Surveys, 28(1):269-271, 1996.
4. http://en.wikipedia.org/wiki/Software_development_process
5. http://en.wikipedia.org/wiki/VModel_(software_development)
6. Jovanovich, D., Dogsa, T., "Comparison of software development models," Published in: 7th International Conference on Telecommunications, 2003.
7. Apoorva Mishra, Deepty Dubey, "A Comparative study of different software development life cycle models in different scenarios" Published in: International Journal of Advance Research in Computer Science and Management Studies, 2013
8. Maglyas, A.; Nikula, U.; Smolander, K.,"Comparison of two models of success prediction in software development projects", Software Engineering Conference (CEE-SECR), 2010 6th Central and Eastern European on 13-15 Oct. 2010, pp. 43-49
9. Sanjana Taya, Shaveta Gupta, "Comparative Analysis of Software Development Life Cycle Models".
10. Vishwas Massey, K.J Satao, "Comparing Various SDLC Models and The New Proposed Model On The Basis Of Available Methodology".
11. Software Engineering - A Precise Approach, Pankaj jalote
12. Software Engineering [Seventh Edition], Ian Sommerville.
13. Software Requirements and Estimation, Swapna Kishore and Rajesh Naik
14. Software Engineering - A Practitioner's Approach, Roger S.Pressman