International Journal of Research Available at https://edupediapublications.org/journals
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 04 Issue 07
June 2017

Software Development Methodologies: A Comparative Study

Sushila &
Raghav Mehra
Department of Computer Science
Bhagwant University, Ajmer (Rajasthan)

ABSTRACT: *Industrial behavior is swiftly altering nowadays and there are more and more complex requirements set on encoding solutions. That place conventional software development methods at the rear and go in front to the need for different approaches. In today's world the present approach is called agile. This paper presents is the process of software development and the diverse methods that are applied to the process. Here in this paper some thoughts about modern research of software are also put in plain words. It plans to set the stage for the formalization of a software development methodology committed to innovation orientated IT projects. Managing software development projects involves techniques and skills that are proprietary to the IT industry. Development phases are the building blocks of any software development methodology so it is important to properly research this aspect. Development phases are defined for every showcased methodology. Software development methodologies are compared by highlighting strengths and weaknesses from the stakeholder's point of view. All in all the research paper is formulated and study direction aimed at formalizing a software development methodology dedicated to innovation orientated IT projects is enunciated.*

KEYWORDS: Software Development, Project Management, Development Methodology

INTRODUCTION: Software methodologies are concerned with the process of creating software. Software engineering is the practice of using selected process techniques to improve the quality of a software development effort. This is based on the assumption, subject to endless debate and supported by patient experience, that a methodical approach to software development results in fewer defects and, therefore, ultimately provides shorter delivery times and better value. The familiar compilation of policy, processes and procedures used by a development team or organization to practice software engineering is called its software development methodology or system development life cycle.

In other words a Software Development Methodology is segregate of

software development work into individual phases containing actions with the objective of better planning and management. It is often well thought-out a subset of the systems development life cycle. The methodology may contain the pre-definition of precise deliverables and artifacts that are created and fulfilled by a project team to develop or maintain an application. General methodologies include waterfall, rapid application, spiral development, iterative and incremental development, development, and prototyping extreme programming and multiplicity types of agile methodology. A number of people believe a life-cycle model is a more general term for a category of methodologies and a software development process a more specific term to refer to a specific process chosen by a specific organization. For example, there are many specific software development processes that fit the spiral life-cycle model.

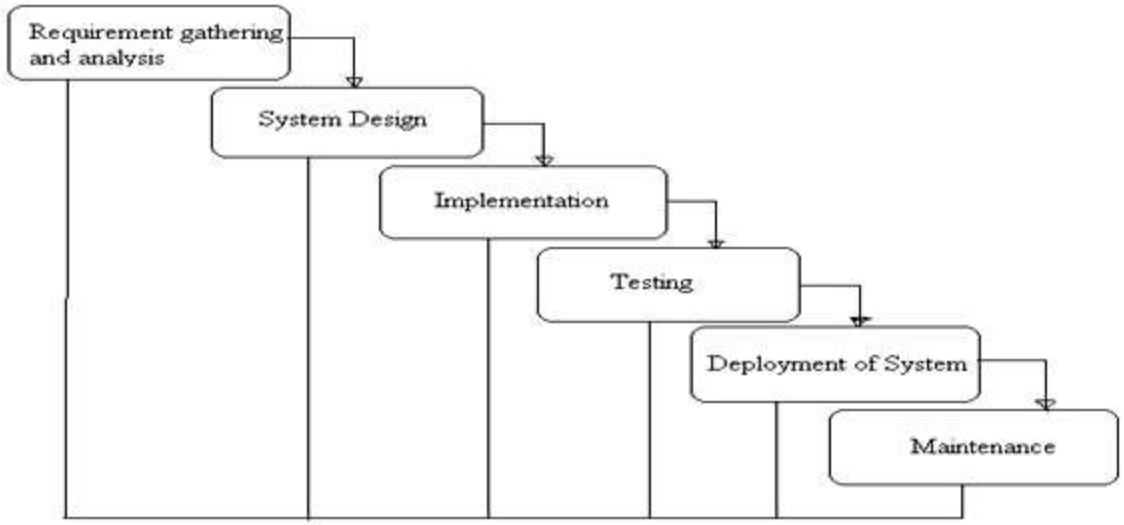
SOFTWARE DEVELOPMENT PROCESS:

A software design process is a construction obligatory on the expansion of a software product. There are quite a lot of methodologies used for such processes; each describing approaches to a range of tasks or behavior that take place during the process. SDLC model is a procedure by which a software project is completed or developed through the distinct processes or stages in present days there are a number of software development models to develop certain software project., so it is very difficult to decide which model is to use for certain type of software project, because every model has its positive and negative aspects, example models take long time to complete, some required less time, some models are for non-technical clients some require well skilled employee, some need crystal clear knowledge about project, one for only developing software from scratch. The achievement and malfunction of the project is also depends upon the selection of definite type of software development model. Just for illustration we can define by very small and inexpensive project, waterfall model is best, some required a lesser amount of human resource, some have time complication matter, for critical assignment projects spiral model is best, for inexpert client prototype model is most outstanding now these days object orient model is well-accepted because it can interact with real world, it does not mean prototype model cannot be used for other projects, it merely means they are good for that type of projects along with other type of projects. Whatever model we choose for developing software projects some mandatory stages are required, without they are as follow:

Waterfall Methodologies: Waterfall methodology was the earliest Process methodology to be set up. It is very simple to

p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 04 Issue 07
June 2017

understand and use. In a waterfall model, each phase must be completed fully before the next phase can begin. This type of software development model is basically used for the project which is small and there are no uncertain requirements. At the end of each phase, a review takes place to determine if the project is on the right path and whether or not to continue or discard the project. In this model software testing starts only after the development is complete. The main advantage of waterfall model is that phases do not overlap.



Advantage: Water fall methodology is trouble-free and straightforward to understand and use. It is effortless to manage due to the rigidity of the model – each phase has specific deliverables and a review process. In this technique segment are processed and completed one at a time therefore phases do not overlap. Waterfall model works well for smaller projects where necessities are very well understood.

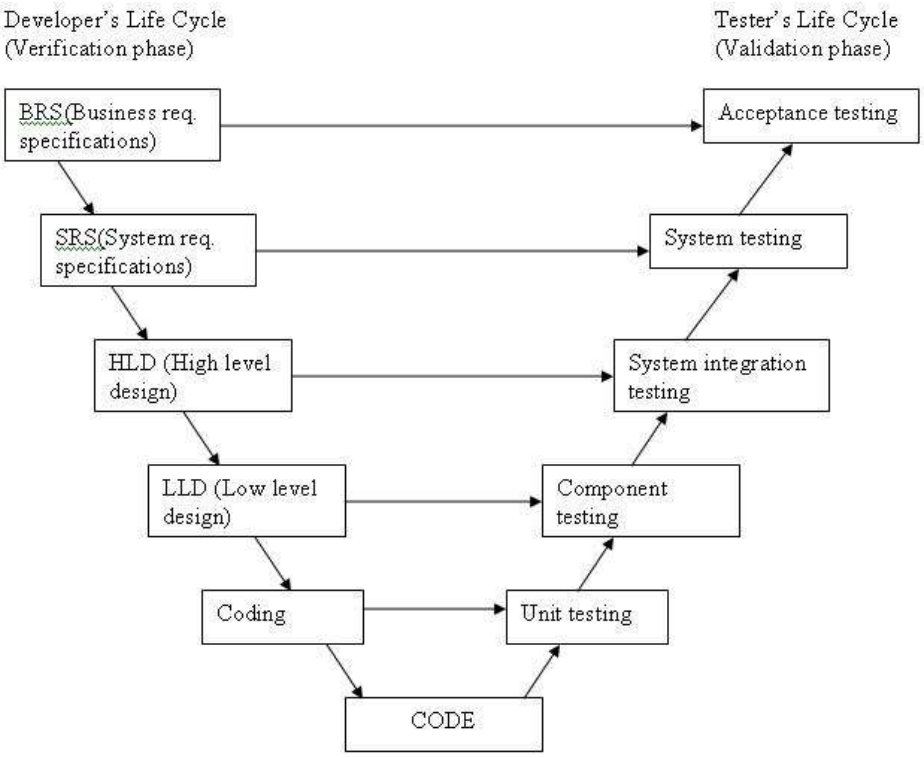
Disadvantage: Everything has some negative aspects so it has some disadvantages viz. on one occasion if a function is in the testing stage, it is very difficult to go reverse and alter something that was not well-thought out in the concept stage. Another drawback of this model is high amounts of risk and uncertainty. It is not a good model for complex and object-oriented projects. Hence it is pitiable model for long and ongoing projects. It is not good for the projects where requirements are at a moderate to high risk of changing.

When to use the waterfall model:

- Thus type of methodology is used when the project is very short.
- This model is bringing into play only when the supplies are very clear and fixed.

- Product characterization is constant.
- Ample resources with required expertise are available freely

V-Shaped Model: V- model means Verification and Validation model. It is an extension of the waterfall technique, Instead of moving down in a linear way, the process steps are bent upwards after the implementation and coding phase, to form the typical V shape. The foremost dissimilarity among v-shaped model and waterfall model is the premature test planning in the v-shaped model. In the V-model, each stage of verification phase has a corresponding stage in the validation phase. These UTPs are executed to eliminate bugs at code level or unit level. Integration analysis process are developed during the architectural intend Phase. These tests verify that units created and tested independently can coexist and communicate among themselves. Test results are shared with customer's team. System Tests Plans are developed during System Design Phase. Unlike Unit and Integration Test Plans, System Test Plans are composed by client's business team. System Test ensures that expectations from application developed are met. The whole application is tested for its functionality, interdependency and communication. System Testing verifies that functional and non-functional requirements have been met. Load and performance testing, stress testing, regression testing, etc., are subsets of system testing.



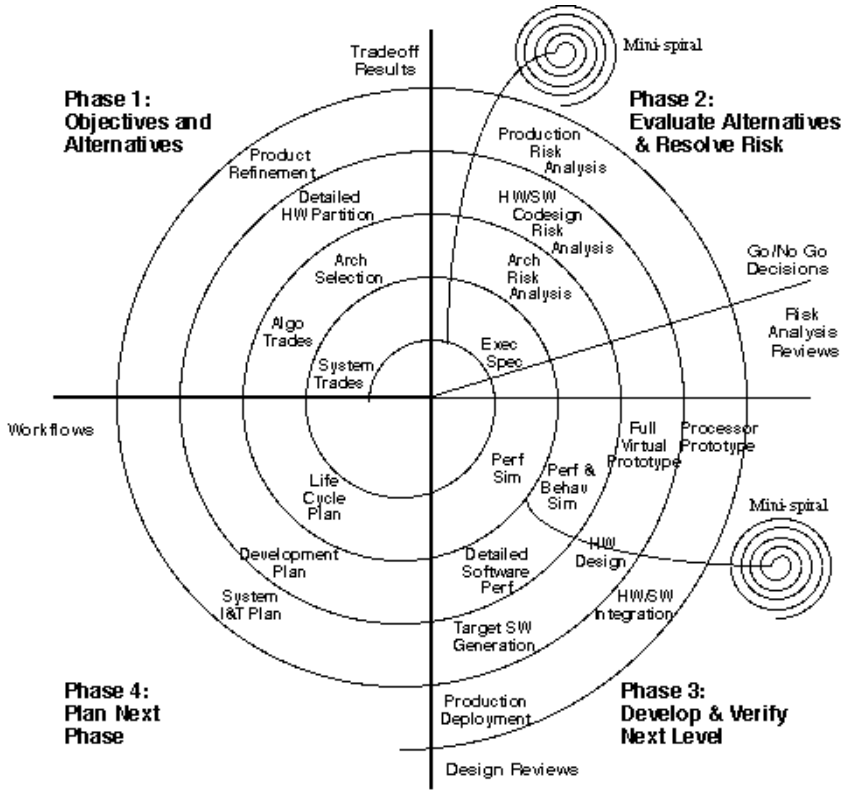
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 04 Issue 07
June 2017

Advantages of V Shaped Model: This method is very much trouble-free and straightforward to use. Testing behavior & actions like planning, test manipulative happens well before coding. This saves plenty of time hence it has higher chance of success over the waterfall model. V shaped model avoids the downward flow of the defects. It Works well for small projects where requirements are easily understood.

Disadvantage of the V Shaped Model: There are few drawbacks in this model like software is developed during the implementation phase, so no early prototypes of the software are produced. If any changes happen in midway, then the test documents along with requirement documents has to be updated. Further it is very unyielding and least flexible.

Spiral Methodology: The spiral model is positioned on risk analysis. It has four phases: Planning, Risk Analysis, Engineering and Evaluation. A software project repeatedly passes through these phases in iterations (called Spirals in this model). The baseline spiral, starting in the planning phase, requirements is gathered and risk is assessed. Each subsequent spiral builds on the baseline spiral. It is one of the software development models like Waterfall, Agile, V-Model. A software project repeatedly passes through these phases in iterations (called Spirals in this model). 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.

p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 04 Issue 07
June 2017

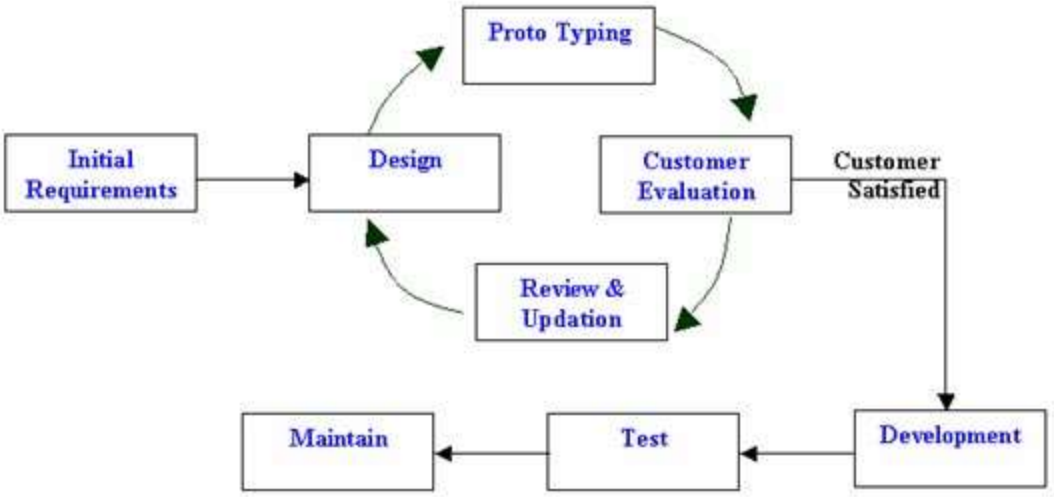


Advantages: In spiral method there is high amount of risk analysis besides it has good for large and mission-Critical projects. In this technique software is produced in the early hours.

Disadvantages: As compare to other techniques it can be a costly model to use. In addition to this risk analysis requires highly specific expertise; therefore, project's success is highly dependent on the risk. All in all we can say it doesn't work well for smaller projects.

Prototype Methodology: Prototyping is a methodology that evolved out of the need to better define specifications and it entails building a demo version of the software product that includes the critical functionality. Initial specifications are defined only to provide sufficient information to build a prototype. The prototype is used to refine specifications as it acts as baseline for communication between project team and project owner. The prototype is not meant to be further developed into the actual software product. Prototypes should be built fast and most of the times they disregard programming best practices. The project owner's feedback is received after the prototype is completed. The Prototyping methodology is suitable for large scale projects where is almost impossible to properly define exhaustive requirements before any actual coding is performed. Prototyping methodology is also suitable for unique or innovative projects where no previous examples exist.

p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 04 Issue 07
June 2017



Proto Type Model

The project owner's feedback is received after the prototype is completed. The Prototyping methodology is suitable for large scale projects where is almost impossible to properly define exhaustive requirements before any actual coding is performed. Prototyping methodology is also suitable for unique or innovative projects where no previous examples exist.

Advantages: There are many advantages using this methodology viz. users are actively involved in the development and since in this methodology a working model of the system is provided, the users get a better understanding of the system being developed. Moreover because of user actively involved errors can be detected much earlier and missing functionality can be identified easily.

Disadvantage: One of the major negative aspects of this methodology is that quicker user feedback is available leading to better solutions. Unfortunately, this methodology may increase the complication of the structure as scope of the system may expand beyond original plans. In extension to this Incomplete application may cause application not to be used as the full system was designed

Agile Software Development (ASD): Agile Software Development is a kind of Incremental model. It is developed in incremental, rapid cycles. In this model outcome is comes in small incremental release with every release structure on previous functionality. Each release is thoroughly tested to ensure software quality is maintained. It is used for time critical applications. Extreme Programming (XP) is currently one of the most well known agile development life cycle model. In software application development, agile software development (ASD) is a methodology for the creative process that anticipates the need for

p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 04 Issue 07
June 2017

flexibility and applies a level of pragmatism into the delivery of the finished product. Agile software development focuses on keeping code simple, testing often, and delivering functional bits of the application as soon as they're ready. The goal of ASD is to build upon small client-approved parts as the project progresses, as opposed to delivering one large application at the end of the project.

Advantages: Agile methodology supports customer fulfillment by rapid, nonstop delivery of useful software. People and interactions are emphasized rather than process and tools. Customers, developers and testers constantly interact with each other. Working software is delivered frequently (weeks rather than months). Confronting each other conversation is the best form of communication. Close daily cooperation between business people and developers. It gives continuous attention to technical excellence and good design and also provides regular adaptation to changing circumstances. Even late changes in requirements are welcomed.

Disadvantage: Working on huge system, it is difficult to assess the effort required at the beginning of the software development life cycle. In this methodology there is lack of emphasis on necessary designing and documentation. The project can easily get taken off track if the customer representative is not clear what final outcome that they want. Only superior programmers are capable of taking the type of verdict required during the maturity process. Hence it has no place for newbie programmers, unless combined with experienced resources.

When to use Agile:

- When new changes are needed to be implemented. The freedom agile gives to change is very important. New changes

can be implemented at very little cost because of the frequency of new increments that are produced.

- To implement a new feature the developers need to lose only the work of a few days, or even only hours, to roll back and implement it.
- Unlike the waterfall model in agile model very limited planning is required to get started with the project. Agile assumes that the end users' needs are ever changing in a dynamic business and IT world. Changes can be discussed and features can be newly effected or removed based on feedback. This effectively gives the customer the finished system they want or need.

Comparison of Methodologies

Methodology	Characteristics	Strengths	Weakness
Waterfall	- Wide-ranging certification - Thorough planning - Linear - Sequential process - Each phase has its own deliverables	- Easy to manage -Easy to understand for the project owner and team	- Working code is delivered late in the project - Does not cope well with changing requirements - Low tolerance for design and planning errors
V- Shape	- Introduces testing at every development stage - Highlights the key safeguarding	-Low bug rate -Easy to be aware of and use	- Vulnerable to scope creep - Relies heavily on the initial specifications
Spiral	- Focuses on objectives, alternates and constrains -Emphasize risk analysis -Evaluates multiple alternatives	-Working code is delivered early in the project. - Minimizes risk	- Costs generated by risk handling - Dependent on accurate risk analysis

Prototyping	-Build one or more demo versions of the software product -Project owner is actively involved - Prototypes are meant to be discarded	- Precise classification of application - Early feedback from the project owner -Improved user skill -Early identification of missing or redundant functionality	- leads to unnecessary -Increase functions complexity -Increased programming effort -Costs generated by building the prototype
Agile Method	-Value driven development -Continuous planning -Relative Estimation -Continuous testing -Continuous improvement -Cross functional terms	- Rapid, incessant delivery - Regular adaptation - constantly interaction - Technical excellence	- Easy to slack and let discipline go. -Peer pressure within the team can be immense. - Does not scale well. - Skill developers

Conclusion: There are a lot of presented models for mounting systems for diverse sizes of projects and requirements. These models were established between 1970 and 1999. Waterfall model and spiral model are used commonly in developing systems. The waterfall, spiral models, prototyping are believed to be those most pertinent to the Department of Defense. An understanding of these models and techniques provides a foundation for understanding others as move further to encounter with them. Each model has advantages and disadvantages for the development of systems, so each model tries to eliminate the disadvantages of the previous model. Basically, Software Development Methodologies follow two major philosophies: heavyweight & lightweight. Heavyweight methodologies are suitable for projects where requirements are unlikely to change and the software complexity allows for detailed planning. Heavyweight methodologies are easy to understand and implement. They provide solid documentation and appeal to project owners because they are well structured and showcase tangible deliverables for every stage of the project. With heavyweight methodologies the project manager can easily perform tracking, evaluation and reporting. The project owner is considerably involved only in the research and planning stages. Lightweight methodologies are suitable for projects were specifications are unclear or are likely to change due to project internal or external factors. Lightweight methodologies

are based on an incremental approach were software is delivered in multiple consecutive iterations, all of them being functional versions of the application. Lightweight methodologies provide great flexibility and can easily adapt to change. They promote early delivery of working code, self-organizing teams and adaptive planning. The project owner is heavily involved in all the stages of the project as its input and feedback is critical for the success of lightweight methodologies. When choosing a software development methodology project owner profile, developer's technical expertise, project complexity, budget and deadlines must be taken into account. Often no methodology will fit perfectly the profile of a specific project. Then the best matching methodology should be used or in the case of experienced project teams and project managers a combination of methodologies could be introduced. In the case of innovative software development projects a new methodology is required. This topic is a subject for further research in the software development field.

REFERENCES:

[1]. M. Davis, E.R.Comer, "A Strategy for Alternative Software Development Life Cycle Models", Journal IEEE Transactions on Software Engineering, Vol. 14, Issue 10, 1988.

- Sharma, B.; Sharma. N, "Software Process Improvement: Analysis of SPI models", Emerging Trends in Engineering and Technology (ICETET), 2009 2nd International Conference on,16-18, 2009.
- Jovanovich, D., Dogsa, T.,"Comparison of software development models," Proceedings of the 7th International Conference on, 11-13 June 2003.
- M. Davis, H. Bersoff, E. R. Comer, "A Strategy for Software Development Life Cycle Models", Journal IEEE Transactions on Software Engineering ,Vol. 14, Issue 10, 1988.
- Apoorva Mishra, "Put Your Process on a Diet", Software Development", IJAR-CS, Vol. 1, Issue 5, Oct. 2013,.
- Sanjana Taya, Shaveta Gupta, "Comparative Analysis of Software Development Life Cycle Models".
- Vishwas Massey, K.J Satao, " Comparing Various SDLC Models And The New Proposed Model On The Basis Of Available Methodology".