International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 3 Issue 11, November-2014

Software Development Process and Methodologies: A Review

Neetu Kumari Lodhi
M.Tech (S.E.) Scholar

Prof. Pankaj Dalal
Associate Professor

Shrinathji Institute of Technology & Engineering, Nathdwara-313301

Abstract **- Software development process and methodologies plays a vital role in the success of software applications. Several software development processes exist for software development. Here we are focusing on two popular development processes, Rational Unified Process (RUP) and Agile Process, with their several methodologies. The Most popular RUP methodologies are Waterfall, Spiral, Prototyping, Rapid application development (RAD), V-shaped etc. and extreme programming (XP), scrum, feature driven development (FDD), adaptive software development (ASD), dynamic system development method (DSDM) etc. of agile process are reviewed with comparison of the various metrics cost, time, complexity, application size, flexibility to change etc..**

Keywords- **Software development process, software development methodologies, Rational Unified Process, Agile process, metrics.**

INTRODUCTION

Software development process is the framework, is used to structure, control and manage the software development. This basic framework allows to the several methodologies to develop and maintain software. The RUP and agile processes are most popular development process in the software industry. These are basic framework that can be modified as per software application requirements. This modification is done by using different software development methodologies. Software development methodologies are collection of procedures, techniques, tools, activities and principles that are used to help to the developer to develop a software application. [01]

2. RATIONAL UNIFIED PROCESS AND ITS METHODOLOGIES

Rational Unified Process is a software development framework that develops a project iteratively and incrementally by using a set of activities such as requirement-gathering, analysis, design, implementation and testing with the help of use-cases. [2]

RUP develops into four phases they are such as Inception, Elaboration, Construction and Transition.

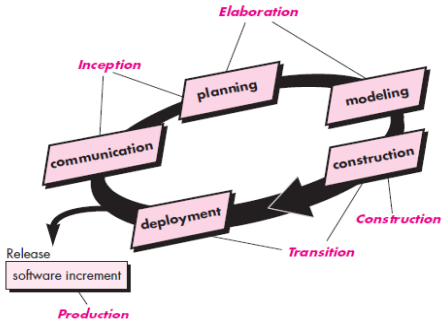

Fig.1 Rational Unified process Phases [03]

The Inception phase includes customer communication and planning. The Elaboration phase includes planning and modeling activities. The construction phase make each use cases operational for end users. The testing and deployment activity are part of transition phase. Deployment includes delivery and feedback activity. [03]

There are three core principles of RUP are use-case driven, Architecture centric, iterative and incremental. Use-case model describes the complete functionality of the system. Architecture presents a view of complete system that control system development. Architecture involves usability, performance, resilience, functionality, reuse and comprehensibility; all within economic and technological constraints. RUP process splits the project into sub parts and develops them iteratively and incrementally. Iterations are steps in workflow and increments are growth in the project. [02]

Methodologies that falls under RUP process are waterfall, spiral, prototyping, rapid application development (RAD), V-shaped, incremental, iterative etc.

Waterfall model was proposed by Winston Royce in 1970. It is linear, sequential and conventional model. It follows requirement gathering, design, coding, testing, delivery and maintenance in sequential manner thus no phase can be started before previous phase has been completed and there is no way to go in previous phase in order to make changes during the development. **Advantages:** Waterfall model is useful where all the requirements can be gathered initially. It has well defined output after each phase. It is easy and simple to understand. **Disadvantages:** It takes long time due to sequential manner, no early prototypes, very rigid. Idealized hence do not match with reality. [04][05]

Spiral model

In order to remove causes of failure of waterfall model many methodologies have been developed that are based on the iterative development. Spiral model combines the iterative and incremental approach in order to offer opportunities to make changes or add new requirements. It focuses on risk analysis during each iteration/spiral. **Advantages:** Risk analysis. It is most suited for complex safety critical system, and early visibility of prototypes. **Disadvantages:** Expensive, consumes long time duration, complex to understand and implementation, requires risk analysts, does not work well for small, less risky projects. [04][05]

Prototyping

Prototyping develops prototypes with main aspects or functionality to ensure to both developer and customer that it is feasible solution for proposed problems. If it is feasible then prototype is refined and turned into final product. **Advantages:** It reduces time and cost, user involvement, change tolerant, quick implementation of incomplete but functional requirements, identifies requirements during development. **Disadvantages:** Insufficient requirement and design analysis, less documentation, schedule, cost and time cannot be estimated. [06][07]

Rapid Application Development (RAD)

RAD model is based on tight time constraints hence used where requirements are well understood with restricted scope. It is most suitable to offer full functionality in short time duration. RAD model make use of techniques and tools in order to fast development. It splits the project for parallel development to reduce complexity and time. **Advantages:** Short time duration, reusability of components, modularization, works well for small or medium size application. **Disadvantages:** Costly due to use of ample resources with experts, tight time constraints, requires skilled team members to make interfacing among the existing components and newly developed components. [06][07]

V-Shaped Model

V-Shaped model is like the waterfall model but start testing from the early phases in order to identify problems in early phases. **Advantages:** Easy and simple to understand and use, each phase has well defined documents or deliverables, high success rate over waterfall model due to testing from the very beginning, Works well where requirements can gather initially, documentation stables the requirements. **Disadvantages:** Sequential hence no phase can be start before the previous phase has been completed, very rigid, no early prototypes are developed due to implementation is done only at last, does not provide clear path for the problems found during testing. [04][08]

3. AGILE PROCESS AND ITS METHODOLOGIES

Agile process is a framework, based on the iterative and incremental development in which requirements are accomplished by the welcome of changes and active user involvement during development. [09]

Agile process follows several phases such as requirements-gathering, analysis, design, coding, testing and delivery of partially implemented product and waits for customer feedback. These phases carried out continuously until customer gets satisfied. Customer satisfaction is the highly prioritized feature with short development time. [10]

Agile process employs various methodologies for different types of application. The most popular agile methodologies are Extreme Programming (XP), Scrum, Feature Driven Development (FDD), Adaptive Software Development (ASD), Dynamic System Development Method (DSDM), Crystal Clear, Lean development etc. Each methodology concerns with best practices in order to achieve maximum benefit.

Extreme Programming (XP)

XP is introduced by Kent Beck in 2000, based on small release in order to satisfy the user about progress of the project. The twelve best practices make it successful. It emphasizes on the simple design, pair programming, continues code improvement, and test based development. Integration started as the small releases have been developed. Kai Stapel, Daniel Lübke, Eric Knauss [11] described that to fulfill the requirements of the software application extreme programming applies twelve best practices: planning game, small releases, metaphor, simple design, testing pair programming, collective code ownership, continues integration, 40-hour week, onsite customer and coding standards. **Advantages:** Short time duration, Small team, user involvement, user can see the progress of the development, test driven development. **Disadvantages:** Onsite customer may be embracing for the development team, pair programming is costly, requires only experienced and skilled team members, works well only with small-medium sized project. [04][08]

Scrum model

Scrum methodology was introduced by Ken Swaber in 1995. Scrum methodology is based on the sprints, daily scrum meetings and sprint planning. Sprints are small functionality that is developed within 30 days. This duration never extends hence customer gets satisfied due to on time delivery. Scrum model resolves the problems and complexities of each team members in daily scrum meeting. **Advantages:** Short time duration, daily scrum meeting, user involvement, review sprints, focuses on team communication and can work with any technology. **Disadvantages:** Works well only with small sized application, requires collocated team members, employs skilled and experienced team. [09][10][12]

Feature Driven Development (FDD)

FDD is based on the feature based decomposition. Features are small client valued functions. FDD emphasizes on planning, upfront design and quality assurance activities. FDD decompose the project into small features and allocate to the small feature team. Feature team develops the feature parallel and owned by specific feature owner who is responsible for the code of the software. Each feature develops in one to two weeks. FDD puts heavy emphasis on the quality assurance activities hence performs inspections, reviews, walkthrough etc. Sadhna Goyal [13] says that FDD applies six best practices to develop a software application successfully: domain object modeling, developing by feature, individual class (code) ownership,

features team, inspections and configuration management. **Advantages:** Ensures quality of the product, costing as per feature, team collaboration, user involvement. **Disadvantages:** Not powerful on small sized application, sometime heavy inspections are embracing activities, long time duration. [03] [12]

Adaptive Software Development (ASD)

ASD is the risk driven approach for the complex projects. It starts risk assessment as early as possible. ASD emphasized on the human interaction and team self organization. It consists in three phase speculation, collaboration and learning. In speculation phase it makes planning for what to achieve in each development cycle. It focus on the interaction between customer and team members to share their knowledge in collaboration phase. It allows taking decision by the team members' instead of formal method of getting permission from the higher authorities. In learning phase it emphasis on learning of individual team members in order to develop a project in more efficient manner. In learning it focuses on three focus group, technical reviews and project postmortems. **Advantages:** Risk driven development, team collaboration, focuses on individual team members learning, quality reviews and time boxing. **Disadvantages:** Over dependence on inter human collaboration, not scalable. [03] [13]

Dynamic System Development Method (DSDM)

DSDM is very dynamic approach. DSDM based on four values and nine principles. Values are individuals, working software, collaboration and responding. These values are supported by the nine principles. It mainly emphasize on the active user involvement, frequent delivery, time-boxing and respond to change. DSDM applies MoSCoW rules for successful development. Acronym for MoSCoW is Must have, should Have, Could have and Want to have. **Advantages:** Eliminate the problems of missing deadlines, going over budget, management not committed, frequent release, time boxing. **Disadvantages:** Unstable requirements, requires on site customer, not suitable for safety critical projects, frequent delivery leads to not to maximizing quality. [03][14]

4. COMPARISON OF METHODOLOGIES

The discuss methodologies are compare with respect to eight metrics. The comparisons are tabulated as under

Table 1. Comparison of methodologies

Rational Unified Process							
	Metrics						
Methodology	Application Size	Known Requirements Initially	Risk and Security	Complexity	Quality	Flexibility to change	Time	Cost
Waterfall	Small-Medium Size	Requires all the requirements initially	Risk may be present due to post implementation	Resolve complexity via modularization and High level design	Good Quality	Very rigid	Long Time Duration	High Cost
Spiral	Large Size	Involves requirements during the development	Remove Risks	Resolves High complexity	High Quality	Flexible	Long Time Duration	Expensive
Prototyping	Small-Medium	Involves requirements during the development	Risk may present due to less time and cost	Don't resolves complexity	Less Quality	More flexible	Short Time Duration	Less cost
Rapid Application Development (RAD)	Small-Medium	Requires most of the requirements initially	Risk may presents due to reusable components and technical issue	Resolves only less complexity	Good Quality	Flexible	Short Time Duration	High Cost
V-Shaped	Small- Large	Requires all the requirements initially	Risk reduction via testing during development but post implementation may carry risks and uncertainty	Resolves complexity via modularization and high level design	Good Quality	Rigid	Long Time Duration	High Cost
Agile Process								

Extreme Programming (XP)	Small-Medium	Involves requirements during the development	Risk reduction via test driven development and reviews	Don't resolves high complexity	Good Quality	More Flexible	Short Time Duration	Low Cost
Scrum	Small-Medium	Involves requirements during the development	Risk reduction via sprint reviews	Resolves complexity	Good Quality	Flexible	Short Time Duration	Low Cost
Feature Development Methodology (FDD)	Medium-Large	Involves requirements during the development	Risk removal via milestones	Resolves complexity	High Quality	Flexible	Long Time Duration	According to Application size
Adaptive Software Development (ASD)	Small- Large	Involves requirements during the development	Risk removal via continues risk updating	Resolves complexity	High Quality	More Flexible	According to Application Size	According to Application needs
Dynamic System Development Method (DSDM)	Small- Large	Involves requirements during the development	Risk removal via continues risk updating and testing	Resolves Complexity	Good Quality	More Flexible	According to Application Size	According to application needs

5. CONCLUSION

This study will make selection process easy and less time consuming and thus very helpful for development team. In this paper we have made comparison of the different software development methodologies on the basis of certain quality metrics size, known requirements initially, risk and security, complexity, quality, flexibility to change time and cost. On the basis of these metrics developer can decide which of these software development methodologies should be suitable for that particular software application. Selecting the appropriate software development methodology improves development process.

REFERENCES

1. Software development methodology, Wikipedia
2. Lorena Azócar Merida, Rational Unified Process (RUP) and its relationship with the end-user in the software system development process, Master thesis, January 2008
3. Roger S. Pressman, "Software Engineering: A PRACTITIONER'S APPROACH, 7th edition", ISBN 978–0– 07–337597–7, Mc Graw Hill Higher Education, 2010. (QA76.758.P75)
4. Nabil Mohammed Ali Munassar and A. Govardhan," A Comparison between Five Models Of Software Engineering", ISSN (Online): 1694-0814, IJCSI International Journal of Computer Science Issues, Vol. 7, Issue 5, September 2010
5. Apoorva Mishra and Deepty Dubey, "A Comparative Study of Different Software Development Life Cycle Models in Different Scenarios", ISSN: 2321-7782, International Journal of Advance Research in Computer Science and Management Studies, Volume 1, Issue 5, October 2013
6. SELECTING A DEVELOPMENT APPROACH, March 27, 2008
7. Ms. Shikha maheshwari and Prof. Dinesh Ch. Jain, "A Comparative Analysis of Different types of Models in Software Development Life Cycle", , ISSN: 2076-734X, EISSN: 2076-7366, International Journal of Advanced Research in Computer Science and Software Engineering, Volume 2, Issue 5, May 2012
8. T Bhuvaneswari1 and S Prabaharan, "A Survey on Software Development Life Cycle Models", ISSN 2320–088X, International Journal of Computer Science and Mobile Computing (IJCSMC), Vol. 2, Issue. 5, May 2013.
9. MALIK HNEIF, SIEW HOCK OW, Review of Agile development Methodologies in Software Development, International Journal of Research and Reviews in Applied Sciences ISSN: 2076-734X, EISSN: 2076-7366 Volume 1, Issue 1(October 2009)
10. Sheetal Sharma, Darothi Sarkar, Divya Gupta, Agile processes and methodologies: A conceptual study, International Journal on Computer Science and Engineering (IJCSE), Vol. 4 No. 05 May 2012
11. Kai Stapel, Daniel Lübke and Eric Knauss, "Best Practices in eXtreme Programming Course Design", 30th International Conference on Software Engineering, 2008.
12. Kaushal Pathak and Anju Saha, "Review of Agile Software Development Methodologies", ISSN: 2277 128X , International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 2, February 2013
13. Malik Qasaimeh, Hossein Mehrfard, Abdelwahab Hamou-Lhadj, "Comparing Agile Software Processes Based on the Software Development Project Requirements", CIMCA 2008, IAWTIC 2008, and ISE 2008.
14. Benjamin J. J. Voigt, "Dynamic System Development Method", 20 January 2004