Agile-Fall Process Flow Model – A Right Candidate for Implementation in Software Development and Testing Processes for Software Organizations

Prakash.V SenthilAnand.N Bhavani.R

Assistant Professor Department of Computer Applications, SASTRA University,Tamilnadu,India

Assistant Professor Department of Computer Applications, SASTRA University,Tamilnadu,India

Assistant Professor, Department of Computer Applications, PRIST University,Tamilnadu,India

ABSTRACT

Software development, an interesting and challenging process follows two contrasting approaches: the Conventional sequential method called the "Waterfall model" and the iterative evolutionary "Agile method". This paper aims at an integration of these two contrasting methodologies to form a new method which can be a potential candidate for implementation in the software development and testing process. This can very well be used widely used in a highly adaptive software environment.

***Keywords*:** Testing, Agile, software testing, process models

1. Introduction

In today's world of rapid software development, companies are reporting a great deal of success in meeting rapidly changing customer needs through the contemporary Agile development method. However, there is also a school of thought that the Agile method only works for small, collocated teams that includes on-site customers. This would impose some major problems if the customers cannot be onsite full-time, or the development team is distributed around the world. Thus, many similar companies are finding that they must employ some traditional development processes like the waterfall, especially on large projects.

Considering this fact in mind this paper compares and contrasts both the Waterfall method and Agile method. It then identifies specific conflicts that companies face along with the suggested strategies to resolve these conflicts to come up with a more robust and excellent integration of traditional and modern software development methodology by suggesting a revised integrated model which combines the benefits of both.

2. Waterfall Method

The Waterfall method is a software development method in which development is seen as flowing steadily downwards through the phases of requirement analysis, design, implementation, testing, integration, and maintenance without going back and revisiting the requirements as in Figure 1. It assumes that all requirements can be accurately gathered at the beginning of the project. In fact, attempts at up-front requirement specification will leave out some important details simply because the stakeholders cannot tell developers everything about the system at the beginning of the project at the same time developers would also not have a proper understanding of a brand new system during the early stages. Consequently, the problems of undefined, changing, and emerging requirements present a very large challenge to Waterfall model projects. Thus the cost of change in a Waterfall project increases exponentially over time because the developer is forced to make any or all project decisions at the beginning of the project. It is worth considering some major disadvantages of the waterfall model at this point. The biggest drawback is we cannot go back if things went wrong in the design phase it will continue to go wrong till the implementation phase.

It is not possible for software companies to release the working model of the software to their clients

until the final stage of the development cycle is complete.

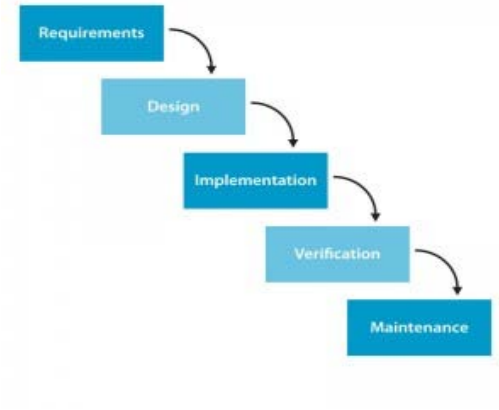

Figure I. The Waterfall Model
Source : http://www.waterfall-model.com

3. Agile Method

Right from its inception in 2001 The Agile software development techniques are gradually being accepted as viable alternatives to traditional software development methodologies like the waterfall model. It leads to better quality software in a shorter period of time.

The Agile method promotes an iterative mechanism for producing software. It increases the iterative nature of the software lifecycle by tightening design-code-test loop to at least once a day as opposed to once per iteration.

Figure II . A SCRUM Process
Source : http://www.ppmstudio.com

In general, the Agile method is a lightweight process that employs short iterative cycles, actively involve users to establish, prioritize, and verify requirements, and rely on a team's tacit knowledge as opposed to documentation. The key concepts of Agile are it's emphasized on

- Individuals and interaction over processes and tools.
- Working software over comprehensive documentation.
- Customer collaboration over contract negotiation.
- Responding to change over following a plan

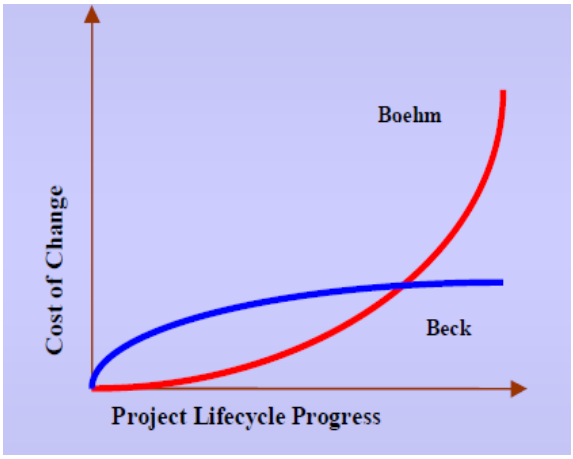

Figure III. Cost of change Curve

Some of the Agile methodologies are Extreme Programming (XP), SCRUM, Crystal and etc. An example of the SCRUM process flows is shown in Figure 2.

Agile visionary Kent Beck challenged the traditional cost of change curve evidenced by Barry Boehm [1] over twenty years ago. Beck's method espouses that the cost of change can be inexpensive even late in the project lifecycle while maintaining or increasing system quality [2] as shown in Figure 3.

4. Agile Success Stories

Calisto had ten simultaneous releases of Eclipse products at the same time[7].It aims to improve the productivity of the developers working on top of Eclipse frameworks by providing a more transparent and predictable development cycle. So, Callisto is remarkable in that it provides a synchronized set of releases to facilitate implementation of Eclipse for developers using them to build their own applications, tools and products. Until now, these different projects have had different release cycles.

Mike Milinkovich, executive director of the Eclipse Foundation since 2004, revealed the reason for this success like this: "Doing your software development transparently has massive advantages. We use agile methods within Eclipse" [7]. We can found lots of success stories from organizations who implements agile methodology.

5. Agile Challenges and The Proposed Agile-fall Solution

Although Agile method seems to be useful, most of the companies have found out that the Agile practices are less burdensome and more in tune on small, standalone projects. There are complexities of scaling

up and integrating them into traditional, top-down systems development organizations.
Below are some of the challenges and recommended resolutions that in-turn invokes the recommended Agile-Fall flow as in Figure VI when we integrate the traditional methods with the Agile development method.

5.1. Early Communication Gaps

Potentially ambiguous requirement statements can be produced as the customer and product developer failed to collaborate sufficiently in the early stages of development. This risk increases in the Agile method because requirements are often written as story cards, which depends highly on face to face communication to resolve different requirement interpretations

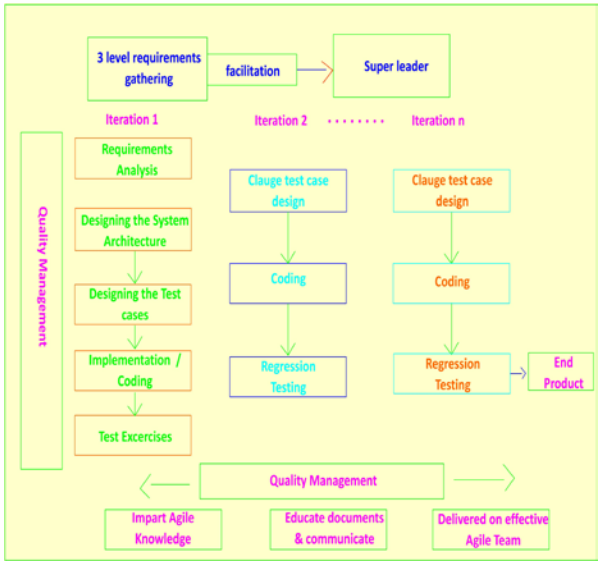

Figure IV. Recommended Agile-fall Process Flow

5.1.1. Three Levels of Requirements. One way the requirements can be controlled is to use three levels of requirements. The first level is the high level user stories that scope the complete project from customer requests. This level includes features which have yet to be fully analyzed. The second levels of requirements are developed collaboratively with customer by clarifying and identifying the detailed work for next iterations. The third level creates a priority list, where the customers themselves are asked to rank the requirements which are very crucial and must be present during the first release. There can still be customer's out-of-scope requests, but agreeable detailed requirements are collaboratively established by customer and product developer for each iteration [4].

5.1.2. Facilitator.

A facilitator is recommended especially for multiple documentation drops. They will participate in team meetings and provide a valuable service by capturing key verbal points and whiteboard sketches, thereby providing useful maintainable lightweight documentation that would help both customer and product developers [5].They must have the coordination skills and knowledge towards the software Development life cycle. The very idea of agile methodologies itself is to reduce the documentation work. Our process flow is not against that core Idea of Agile, so only light weight documents are prepared.

5.2. Large and Distributed Projects

On large distributed projects, customer cannot be on-site full time. Lack of communication with customer may breakdown the progress for large and distributed projects.

5.2.1. Super Leader:

A super leader has domain experience on a project and can serve as a customer communicator. The role of a super leader is to answer questions quickly to maintain the velocity of the Agile team. If the super leader does not know the answer, he will know who to call to get the answer from. The super leader is accountable in building a strong and continuous relationship with customers who are not on-site by always striving to use the most effective communication channels. Therefore, physical location and project size are not an excuse for lack of communication.

5.3. Issues of Conflicts

Conflicts occur on two issues. The first is on how to extend, evolve or interoperate the Agile processes. The second is on how to merge Agile with traditional development methods without either killing agility or undermining the years spent defining and refining the systems and software engineering processes [6].

5.3.1. Need for Documentation and Communication

In order to know how to map Agile to traditional method and how compatible they are, documentation and presentation are needed to clearly define and communicate the process to all parties. After significant analysis on existing and proposed processes, document the key terms, roles and responsibilities, as well as architecture that support the compartmentalization of Agile and traditional method.
Traditional milestone redefinition to fit an iterative approach, Agile practices that supports existing processes as well as risk evaluation from the project management's perspective is also noted down. The aim is to present this tailored Agile document to the customer and development team members to raise their awareness on the risk and solution for merging both methods.

5.4. Conflicts with the Big P – People

Adoption and integration of Agile method can cause people conflicts. Paradigm shift in the process may not be accepted by team members and customers. Teams may refuse to use new development methods with pair programming and shared ownerships [7].

5.4.1. Knowledge Base and Agile Team Building.

In fact, much of the paradigm change is actually aimed at empowering individuals by supporting reasonable goals, shorter feedback cycles, ownership and flexibility.
The Agile method places more emphasis on team competency in the project. In order to integrate Agile method into traditional method, knowledge base plays an important role in raising the knowledge bar of team members. Knowledge base or knowledge repository is especially useful for large projects and it is an inexpensive way to increase the expertise of all team members.
Besides that, the Agile method requires a team to have a common focus, mutual trust and respect, and a collaborative, but speedy decision making process. People who work together with good communication and interaction can operate at a noticeably higher productivity than the talented people who work individually. Therefore, team building activities are very important in improving the collaboration, communication and good relationship between team members.

6. Conclusion

For companies with geographically distributed teams, diverse cultures and experiences an Integration of Agile and traditional development method is the next step Having better collaboration, communication and concise light weight documentation will allow more rapid integration and eliminate the repetition of ineffective approaches. Facilitator, Super leader, knowledge sharing and team building are the vital keys to foster the effectiveness of a development project. Table 2 shows the comparison between the Waterfall, Agile and the recommended agile-fall Process Flow.

7. References

[1] Barry Boehm. 1981. *Software Engineering Economics*. Prentice Hall, PTR.

[2] Kent Beck. 2000. *Extreme Programming Explained: Embrace Change*. Addition-Wesley.

[3] http://compass.mot.com/go/Agilemoto

[4] Paul E.McMohan. 2005. Extending Agile Methods: A Distributed Project and Organizational Improvement Perspective

[5] Paul E.McMohan. 2004. *Bridging Agile and Traditional Development Methods: A Project Management Perspective*

[6] Barry Boehm, Richard Turner. 2005. *Management Challenges to Implementing Agile Processes in Traditional Development Organization*

[7] Barry Boehm, Richard Turner. 2004. *Balancing Agility and Discipline: A Guide for the Perplexed*

[8] Deborah Hartmann preuss, **June 30, 2006.** "Eclipse "Callisto" an Agile Success Story", http://www.infoq.com/news/Eclipse-Callisto-Agile-Success

Author Profiles:

Prakash is a highly effective teacher and Assistant Professor SASTRA University, Tanjore; Tamilnadu India He has more than 13 years of experience in computer science education. Prakash has a special interest towards software testing and is working towards gaining a Ph.D in Software Testing.

Senthil Anand has got over 13 years of teaching and research experience and currently working as an Assistant professor in SASTRA University,Tanjore Tamilnadu. His areas of interests are Digital Image Processing and Software Engineering.

Bhavani Prakash has got her Master's degree in , Computer Science and Engineering and she has got a teaching experience of about 10 years training students in data mining, compiler design and software testing Working as an assistant Professor in PRIST University, Tanjore, Tamilnadu,India

Criteria	Waterfall	Agile	Agile-fall Process Flow
Understanding of Requirement by Development Team	Low because all the Requirements are gathered up-front	Low Ambiguity in story cards	High -Facilitator
Levels of Customer satisfaction and Product Quality	Low Customer involvement will take place only towards the beginning and in the end of the product development. provide customer with progress visibility during product development	High For small projects with onsite customers	Higher With super Leader and effective communication methods
Product Flexibility to changing business need	Low -Predictive method focus on planning the future in detail	High -Adaptive method that focus on adapting quickly to changing realities -resource loading to Support rapid pace.	Higher -3 level Requirement -involved customer collaboration and HR to resolve resource loading problems
Growth in Team Competency	Low -follow the processes and tools in the provided document	High -no knowledge sharing through knowledge base	Higher -inexpensive way of knowledge sharing through knowledge base
Level of trust between team members	Low - more focus on processes and tools	High - more focus on individuals and interaction	Higher -stress on people strategy such as: team building activities and team motivation

Table 2. Software Development models – A comparison