A comparative study on usage of traditional and agile software development methodologies in software industry of Asia

Syed Faisal Ahmed Bukhari and Hira Khan
Department of Computer Engineering, Sir Syed University of Engineering & Technology, University Road, Karachi-75300, PAKISTAN

Abstract - In recent years, agile methodology has been adopted for software development in comparison with traditional models like Waterfall model, Spiral model, Rapid Application Development (RAD) model. In this work, a comprehensive study has been done for comparison between agile and traditional models and their usage in software industry of Asia. This research is conducted to assist software development professionals for selection of suitable development model for small, medium and large size projects considering factors like scope, quality, cost, time and risk. Based on collected data, it is concluded that for small scale projects, software professionals feel more satisfied in using traditional methodology whereas for medium scale and large scale projects, professionals prefer to adopt agile models.

Keywords Agile methodology, system development life cycle, project scope

1 Introduction

The systems development life cycle (SDLC) is a conceptual model used in the development of project that describes the stages including requirements gathering, designing, implementation / coding, testing and deployment. According to literature (Khurana and Gupta 2012), SDLC methodologies are the process to assure that software meet up established requirements. These methodologies entail the discipline to the development process to formulate the development more efficiently.

These methodologies resolved the problems arising from code and fix strategy. By the time of system growth it becomes increasingly complicated to add new features or to fix any bug (Kamel et al. 2010). By the invention of Waterfall model (the basic SDLC model) changing requirements were fixed once but practically requirements just could not be fixed (Royce 1970).

After requirement gathering, development teams work together with each other to create the best possible architecture for the product. The programmers implement design in code and lastly, the complete designed system is tested and dispatched. This process sounds good theoretically but practically it does not always work well if users change their minds after months or even years of requirements accumulation. This results in building replicates as it is very complex to interrupt the momentum of the project to accommodate the change.

It was argued (Beck 1999) for the solution of the above issues that the iterative and incremental techniques break the development into portions. Incremental development intended to reduce development time by various overlapping increments. Iterative development process breaks the project into deliverable iterations of variable length. Similarly, the Spiral Model evades detailing and defining the whole system upfront contrasting iterative development, where the system is built into pieces and prioritized by means of functionality. Spiral and iterative development process models presented a great increase in agility over the Waterfall process, but some practitioners believed that they still did not take action for change in the growing business world.

2 Research objectives

The overall purpose of this research is to conduct a detailed review of both traditional and agile methodologies applied in software industry of Asia. The main focus of this research is to study adoptability of methods and compare them in order to attain professional satisfaction. Because there are many developers, who have implemented the agile methodologies, but on the other hand, there are several developers who are satisfied with traditional methodologies according to their company’s trends and needs. However, the research is conducted with the interest of finding empirical data to come up with tabular plans that would be useful for software industry.

The primary aim of this research is to study and contribute the knowledge about adopting the newly introduced agile software development model in contrast of SDLC models. Specifically, the research objectives are summarized in threefold:
1) To compare characteristics, strength and weaknesses of traditional SDLC models and agile models.

2) To gain an understanding of the reasons that why information technology (IT) industry are now inclined on agile models and which software development method (agile/traditional) is more suitable for software development professionals (with respect to scope, quality, cost, time, and risk) on the basis of three different scales?

3) To discuss risks associated with each of these methodologies.

A questionnaire was designed to get the opinion of software developers in Asia to evaluate usage of methodologies for software development according to the size of the project.

2.1 Traditional heavyweight models

Software methodologies like waterfall model, spiral model and RUP model etc. are often called traditional software development methodologies and classified as heavyweight methodologies (Nikiforova 2009). These are based on sequential series of steps that use comprehensive description and heavy documentation for all the sets of requirements and they do not support requirement changes. For example, waterfall model has linear nature and it is easy to follow and implement (Petersen et al. 2009). Its disadvantage is that it forces to define requirements thoroughly during the system requirements definition stage which is unrealistic (Chocano 1996).

The spiral model was revealed by Barry Boehm in 1988. In this model, although the developers and customers better recognize and respond to risk at every evolutionary level. The drawback of this model is that if a major risk is not revealed and handled, problems will occur with the possibilities of entire software failure. Another traditional heavyweight model is RUP model which was proposed by Rational Unified Corporation. It is an incremental process where the overall project is divided into iterations (Booch et al. 1998). According to researchers (Amlani 2012), this is a complete methodology and all of its documentation is easily available. The drawback as mentioned by programmers is that the process is cost consuming and its lengthy documentation (which is necessary in all iterations) consumes more time (Runeson and Greberg 2004).

2.2 Agile lightweight models

The most clearly focused description of agility is that it is the aptitude to both produce and retort to change with the intention of profit in an unstable business environment (Highsmith 2002). XP, stands for extreme programming, is most extensively used method in agile methodologies (Beck 1999). It focuses on the development rather than executive and managerial sides of software projects. XP projects begin with a release planning phase, pursued by several iterations, each of which terminates with user acceptance testing (Abrahamsson and Ronkainen 2002). The main focus of XP is to get the job done. The main disadvantage of this technique is that as XP is code oriented rather than design oriented therefore it has less documentation. The lack of formalism and design in this model can be problematic for large programs especially when many team members are associated with the project.

Among all of the agile methodologies, Scrum is exceptional because it initiates the idea of empirical process control. Empiricism states that knowledge comes from experience and decision making based on what is known. Like XP, Scrum is also an iterative and incremental approach. This was started as a framework that has been used to manage difficult products. The most common reason for the failure of project using Scrum is that many professionals are still unfamiliar with Scrum, even after taking a Scrum class or reading a few research papers about it. Sometimes organization is not yet setup the Scrum or may be the teams do not know how to employ scrum according to the company’s present constraints or may be the project becomes too complicated to get it under control.

Feature-Driven Development Process (FDD) is one of the agile processes that do no talk and write too much and unlike XP and Scrum, FDD designed to work with a large team for large projects. FDD splits the large team into smaller feature-focused teams. Unlike other agile methodologies, FDD is more appropriate for large projects. The limitation of FDD is that it does not identify what technology to use. Another disadvantage of FDD is class ownership.

3. Research approach

Commonly two approaches of research methods are used, i.e. quantitative and qualitative. In this work quantitative approach is used. For this purpose a self-administered close ended questionnaire is circulated by using http://www.docs.google.com in order to collect the primary data for statistical analysis. The main purpose for using that platform is to gather data from all over Asia.

Finding relevant professionals and software organizations is a very crucial part for this research. To make sure that survey questionnaire is distributed to the professionals using traditional SDLC and agile methodologies throughout the Asia, much efforts has been done which included access local software houses through their web sites and using www.linkedin.com and www.facebook.com/sqlportal (social networks) to get the maximum number of responses.
3.1 Hypothesis

The aim of this research is to compare the traditional and agile methodologies that the IT industry of Asia prefers for the development of three different scales of projects. For checking whether IT companies are using traditional or agile approach the following hypothesis is used:

H10: Companies do not prefer to work under agile approach rather than traditional approach for small-scale projects.

H1: Companies prefer to work under agile approach rather than traditional approach for small-scale projects.

H20: Companies do not prefer to work under agile approach rather than traditional approach for medium-scale projects.

H2: Companies prefer to work under agile approach rather than traditional approach for medium-scale projects.

H30: Companies do not prefer to work under agile approach rather than traditional approach for large-scale projects.

H3: Companies prefer to work under agile approach rather than traditional approach for large-scale projects.

H40: For small-scale projects, software professionals do not prefer agile (XP, Scrum, FDD) technique to adopt for a successful completion of a product in terms of product’s scope, quality, cost, time and risk.

H4: For small-scale projects, software professionals prefer agile (XP, Scrum, FDD) technique to adopt for a successful completion of a project in terms of product’s scope, quality, cost, time and risk.

H50: For medium-scale projects, software professionals do not prefer agile (XP, Scrum, FDD) technique to adopt for a successful completion of a project in terms of product’s scope, quality, cost, time and risk.

H5: For medium-scale projects, software professionals prefer agile (XP, Scrum, FDD) technique to adopt for a successful completion of a project in terms of product’s scope, quality, cost, time and risk.

H60: For large-scale projects, software professionals do not prefer agile (XP, Scrum, FDD) technique to adopt for a successful completion of a product in terms of product’s scope, quality, cost, time and risk.

H6: For large-scale projects, software professionals prefer agile (XP, Scrum, FDD) technique to adopt for a successful completion of a project in terms of product’s scope, quality, cost, time and risk.

In above hypothesis, Hn0 and Hn (where n = 1 to 6) show null hypothesis and alternate hypothesis respectively.

3.2 Questionnaire formulation

In order to obtain data from IT organizations in Asia, a survey questionnaire was presented to the respondents. This questionnaire was filled by the various employees from IT departments of the companies or software houses. The questionnaire was based on elements of existing project development approach according to three different sizes of the project. It is divided into three parts. The first part contains questions regarding size of project and the methodology used. The second part was related to resources and attributes. The third part was designed to collect data about different characteristics including strengths, weaknesses and risk factors.

4 Results and analysis

As a result of survey, 59 responses were received from different type of software development professional working in all over Asia. After reviewing the results, it was evident that 62.7% of population using agile software process while 37.3% using traditional software development methods (Figure 1). On the basis of this result, hypothesis H1 is rejected while two hypotheses (H2 and H3) stand out as accepted.

![Figure 1: Use of traditional/agile models based on project size](image-url)
terms of five different factors. Out of these six models, three were selected from agile and three from traditional techniques.

Since the test is applied on multiple models in terms of multiple factors, therefore the average values for small (Tsmall), medium (Tmedium) and large-scale (Tlarge) projects are computed by adding up the responses against three models (Waterfall, Spiral and RUP) of traditional technique in terms of all five attributes.

Similarly, the average values for small (Asmall), medium (Amedium) and large-scale (Alarge) projects are computed by adding up the responses against three models (XP, Scrum and FDD) of agile technique in terms of all five attributes.

### 4.1 Small scale projects analysis

On the basis of above procedure, the average resulted percentage for traditional technique Tsmall and Asmall based on five attributes, i.e. scope (TSs = 50% and ASs = 50%), quality (TQs = 50% and AQs = 50%), cost (TCs = 63% and ACs = 38%), time (TTs = 63% and ATs = 38%) and risk (TRs = 75% and ARs = 25%) are 60% and 40% respectively.

\[
T_{small} = \frac{(TS_{s} + TQ_{s} + TC_{s} + TT_{s} + TR_{s})}{5} \quad (i)
\]

\[
A_{small} = \frac{(AS_{s} + AQ_{s} + AC_{s} + AT_{s} + AR_{s})}{5} \quad (ii)
\]

As explained by the results and analysis of above five aspects (that are product’s scope, quality, cost, risk and time); agile doesn’t prove to be the most preferred software development technique used by most of the software professionals for small-scale projects. Therefore hypothesis H40 is accepted and H4 is rejected.

### 4.2 Medium scale projects analysis

Similar to the above approach, the average resulted percentage for traditional technique Tmedium and Amedium based on five attributes, i.e. scope (TSm = 43% and ASm = 57%), quality (TQm = 39% and AQm = 61%), cost (TCm = 35% and ACm = 65%), time (TTm = 35% and ATm = 65%) and risk (TRm = 35% and ARm = 65%) are 37.4% and 62.6% respectively.

\[
T_{medium} = \frac{(TS_{m} + TQ_{m} + TC_{m} + TT_{m} + TR_{m})}{5} \quad (iii)
\]

\[
A_{medium} = \frac{(AS_{m} + AQ_{m} + AC_{m} + AT_{m} + AR_{m})}{5} \quad (iv)
\]

As explained by the results and analysis of above five aspects (that are product’s scope, quality, cost, risk and time); agile is proved to be the most preferred software development technique used by most of the software professionals for medium-scale projects. Therefore hypothesis H5 is accepted.

### 4.3 Large scale projects analysis

For large-scale project, the average resulted percentage for traditional technique Tlarge and Alarge based on five attributes, i.e. scope (TSl = 18% and ASl = 82%), quality (TQl = 29% and AQl = 71%), cost (TCl = 21% and ACl = 79%), time (TTl = 21% and ATl = 79%) and risk (TRl = 21% and ARl = 79%) are 22% and 78% respectively.

\[
T_{large} = \frac{(TS_{l} + TQ_{l} + TC_{l} + TT_{l} + TR_{l})}{5} \quad (v)
\]

\[
A_{large} = \frac{(AS_{l} + AQ_{l} + AC_{l} + AT_{l} + AR_{l})}{5} \quad (vi)
\]

Therefore it is concluded that for large scale projects hypothesis H6 is accepted.

### 4.4 Hypotheses assessment summary

On the basis of above analysis (shown in Figure 1) hypothesis H1 has been rejected while two hypotheses (H2 and H3) stand out accepted. The analysis of five attributes has indicated (Figure 2-6) that hypothesis H4 is rejected while two hypotheses H5 and H6 stand out accepted.

### 4.5 Selection of methodology based on different priorities

During this research the following results are also obtained based on different priorities of software customers:

#### 4.5.1 Strictly follow a project plan

It is clearly mentioned in Figure 7 and Table 1 that approximately 63% of the respondents consider that fulfillment of the user requirement is more important than strictly following an initial plan, while around 25% were against it. The remaining 12% of the respondents were neutral.

#### 4.5.2 Detailed documentation

Heavy documentation seems to be a negative aspect for most of the respondents as shown in Figure 7. About 59% selected it as a bad or slightly bad attribute in the successful completion of the projects, while 27% respondents are agreed with this attribute and 14% are neutral.

#### 4.5.3 Hiring highly skilled professionals

Approximately 81% respondents are agreed for hiring skillful workers to produce good quality software, only 10% respondents considered it as a bad approach while 9% are neutral.
4.5.4 Autocratic management style

Unlike the results of other research works conducted in different countries, 63% of the respondents of Asia like the democratic management style rather than the self-organizing team, 24% respond neutrally and only 13% respondents think that this type of management does not allow the developers to be more creative.

4.5.5 Flexible project plan

Approximately 60% of the respondents think that there should be flexibility in a project plan in order to welcome change in requirements and try to produce software which holds complete sets of user specifications whereas approx 25% of the respondents did not support any flexibility in a project plan while approx 15% responds neutrally.

4.5.6 Focus on working software rather than detailed documentation

As shown in the graph that contrary of the results of detailed documentation, approx 58% of the respondents considered that basic documentation could be less time consuming and will give a positive impact for the successful completion of a project. Approx 27% dislikes this characteristic while 15% of the respondents are impartial.

4.5.7 Use of a good development tool

Approximately 59% of the respondents are not in favor of just using a good development tools and not pay attention in having a skilled professionals for the coding, testing or other development procedures. 24% of respondents considered that using a good development tools are more appropriate rather than hiring a skilled professionals which is costly too. And remaining 17% are impartial.

4.5.8 On-site customer

Having customer on-site is the main attribute of agile methodologies and according to the results as shown in Table 1, the working professionals are not very much satisfy with this attribute as approx 56% of the respondents selected it as a bad or slightly bad for the success of the project while 36% are in favor and 8% are neutral.

4.5.9 Self-organizing team

Approx 61% of the respondents didn’t agree with this attribute because of the reason of results drawn in autocratic management style which is more likely to be followed for the development of the software. Only 27% of the respondents considered self-organized teams as a good attribute and remaining 12% are impartial about this aspect.

![Figure 2: Use of traditional Vs. agile model based on scope](image-url)
Figure 3: Use of traditional Vs. agile model based on quality

Figure 4: Use of traditional Vs. agile model based on cost

Figure 5: Use of traditional Vs. agile model based on risk
5 Conclusion

As a result of this research it is suggested that a hybrid methodology should be adopted by the software organizations working in Asia. It comprises of the following steps:

1. Prepare a flexible plan starting through requirements gathering phase which can be repeated at any stage of development according to the needs of customer.

2. Detailed documentation should be avoided at every stage and step perhaps only imperative and significant documentation must be needed.

3. Don’t completely emphasize on development tools with novice professionals, who don’t have any previous experience and can work on cheap rates. Management must hire highly skilled professionals with novices to deliver a good quality software that meet customer requirements.

4. Self-organizing behavior is not appropriate for the environment of Asia. There should be some team lead decisions for the distribution of responsibilities within a team. The management must allocate the duties to every team member according to their expertise.
5. Customer should always welcome with the set of their requirements but the presence of customer on site is not a good idea for team members.

6 References


Acknowledgements

The authors would like to thank Higher Education Commission, Pakistan for supporting this research.