International Academic Institute for Science and Technology

International Academic Journal of Innovative Research
Vol. 1, No. 1, 2014, pp. 1-5.
ISSN 2454-390X

International Academic Journal of Innovative Research

www.iaiest.com

Development of umbrella activities in agile methodologies

Hooman Kashanian[a] , Mohammad Hassan Peashdad[b] , Mahdi AliPour Kondori[c]

[a] *Department of computer, ferdows Branch, Islamic Azad University, ferdows, Iran.*

[b] *Department of computer, ferdows Branch, Islamic Azad University, ferdows, Iran.*

[c] *Department of computer, ferdows Branch, Islamic Azad University, ferdows, Iran..*

Abstract

Agile software development is a group of software development methods based on repetition in gradual form, in which solutions are applied through self-organization and cooperation among various teams. This comparative planning technique improves the development and evolutional reformation as well as repetitive packaging time. It also enforces quick and flexible changing responses. In fact, quickening is a conceptual framework that improves the prediction of interactions along development recycle. There are several characteristics for these methods that mainly improve development, team work, cooperation, and harmony of process in project life recycle. Agile methods break assignments into short steps with the minimum amount of planning which do not directly deal with long-term planning. Repetitions are short-term frames that usually last one to four weeks. Each repetition bears a reactive practical team in all missions: demands analysis, design, coding, test unit, and passing the test. Finally, a work product is represented to the benefit takers in the end of each repetition. It allows the project adapt with the changes quickly. In an agile project the team is typically a reactive and self-organized application. Team members are often in charge of responsibilities. They separately decide how to deal with the demands of a repetition. Software creation methodologies are frameworks for applying software engineering activities with the aim of creating software systems and include software creating processes and modelling languages. The position of umbrella activities in software creation methodologies includes three periods: first generation methodologies, which concentrated on creation activities but did not cover umbrella ones, and second and third generations of methodologies that fully covered creation activities and took the benefit of project management standard frameworks to perform umbrella activities, and also agile methodologies, which are focused on umbrella activities with proposing a novel framework for software creation.

Keywords: quickening, software development, umbrella activities

Introduction:

Considering ever growing of internet users all around the world, software demands are expanding as well. Due to technologies changes and innovations in operating systems, the demands of users for creating and updating the software are changing in an uncontrolled manner. Moreover, creation and development of software is naturally an irregular and extremely chaos process. To regulate the process, software development methodologies were introduced. Numerous approaches have been introduced for software development in recent 25 years, even though very few numbers of which are used these days. Making use of these methodologies, which practice the development process based on a general fundamental plan, drew great attentions. These approaches were installed on the basis of precise timing as well as design and planning based development so that they were called weighted methodologies. Development methodologies defined traditional information system as an essential scenario to illustrate an image of the control process or the generation of a symbolic status.

This changeability causes much many troubles for heavy approaches, since these approaches practice according to an initial plan predicting the demends in advance. Changing demands is interpreted as changes in the performed timing, activities priorities, applied financial strategies, and in general, as changes in foundation and basis of the practiced planning. Accomplishment of such change will be an expensive and time-consuming process. Due to the aforementioned reasons, it seems undoubtedly vital in the current highly dynamic and changeable world to create new cheap and easy to transfer approaches that their absence is obviously felt. In early 2000 it was declared that the present methodologies at that time were not efficient in several cases and did not meet the desired expected stuffs. As a result software developers conducted enormous researches to find a novel approach for software development which is expected to be cheap without extra weights, which are observed through heavy documentations. It was a background for innovation of quick transfer approaches and software development. Therefore, in the late 90s was established the agile mentality, including a set of values and principles for the development of efficient software by self-organized teams. Topic of new quick transferring approaches became a controversial debate amongst all software communities. At last, the first so called XP (Extreme Programming) software quick transferring approach was introduced. The method was invented by Kent Beck in 1999 and attracted so much attention as the starting point of quick transferring approaches. Since XP approach was enormously welcome, researchers conducted researches and studies in this domain which led to the invention of new quick transferring methodologies one after the other.

Agility is the ability to create and respond to changes in order to make profits in the profession turbulent environment. It has a close relation with the matter of change, which is a result of environmental reformation of the profession and makes organizations to react. If an organization is not able or does not want to show an appropriate response to the changes, it must expect serious damages and eventually death. However, it is interesting that organizations with sufficient capability and agility required to deal with changes are themselves able to generate changes in market and profession environment so that make the situation tougher for their competitors and then make more profit. Agility here means one's ability to respond the changes is stronger than their ability to program and propose plans. However, it does not mean that agility considering methods do not make plans, yet the programming is flexible enough to encounter to changes and make profits.

Agile declaration

This declaration refers to four principles. Egocentrism and interaction prior to processes and tools, applicable software prior to conceptual documentaries, cooperation with customers prior to contract-oriented negotiations, and responding the change prior to imitating the plan. The first one considers reliance on individuals' knowledge, and abilities and interactions among them, and criticizes too much emphasis on processes, plans, and tools.

The second principle notifies more attention to software creation than documentation. Documents are required to train users and to almost to preserve the system. Even though, do not forget the main goal is to develop software not documents.
In the third principle, a problem is noted that software developers are permanently dealing with. It is the lack of common understanding as well as inappropriate cooperation. It is vital for a project to make a deal, but not enough to communicate with customers. In software development continuous cooperation with the customers is a demand.
The last principle is referring to responding the changes. Customer's requests from software changes, so does the professional environment to which the software is dependent. And also passing the time technology is changed. Change is a fact in software development and must be supported by development process. Not only having plan for the project is not a mistake, but also it is worrying not to have it. However, the project plan should be facilitated and capable of dealing with the probable changes.

The proposed method

In the proposed approach in this research, in order to develop agile project management in software creation processes, the agile methodologies are evaluated first and then umbrella activities are separated from agile processes. Finally, a general perfect framework is introduced for agile project management. Method engineering as one of the most significant branches of software process engineering, matches and produces software creation processes according to the certain properties existing in each project. The method includes three strategies on the basis of pattern, prevention, and development. Moreover, process patterns are composed of three parameters including phase, stage, and task.
To explain the proposed approach, first of all we have a general review on the protocol and principles of agility. Better software creation approaches are recognized in agility protocol. Flowingly, the team practices based on them and recommend others to do so at the same time. So values are yielded such as individuals and their interactions rather than processes and tools, or applicable software rather heavy documents, etc. while the former values in the mentioned cases are more significant, more value is put on the latter ones.
One can express some of the agile principles as the following way including the fact that the first priority in doing projects is to satisfy customers through quick and permanent delivery of the software, responding demands changes, delivering applicable software in a short period of time, continuous and persistent collaboration with the end users, taking the benefit of capable and skilled people in team, supporting agile processes of coordinate and stable production, and so on. Now, according to the general properties mentioned in agility protocol and principles, the proposed framework in this study presents agile methodology process patterns which are composed of two APM3 metamodels and APM framework.
APM3 metamodel (Agile Project Management Methodology Metamodel) makes models of project management processes by separating them into composing elements. It also introduces the structural and behavioral model of agile project management processes plus a model abstracted from the agile project management processes. The structure of APM3 metamodel is composed of elements like framework, which includes process framework, project elements, some goals and limitations, and environment elements including target environment, project, and creation.
Although APM (Agile Project Management) framework is defined based on APM3 modeling and project management processes illustrate enormous elements in different levels. Inputs in APM framework are engaged with functions like process groups, main processes, tasks, activities, patterns, and techniques and then move to exit.

Combination of agility and architecture

Three strategies are possible to be followed. Using architecture in agile approaches, quickening architecture recycle work, and injecting agility and architecture in a novel method.

Making use of architecture in agile approaches

Agile approaches do not typically consider a so called architecture product in their development process. Consequently, there is no report of activities such as design, documentation, and architectural evaluation.

In some approaches, there are cases approximately similar to the system architecture. Here are three cases of them: metaphor in XP, superior model in FDD, and high level design in scrum. Metaphor is one of the XP approaches that tries to provide a tale-like easy to understand picture for all the benefit takers so that they are able to communicate each other on the basis of the picture. Since metaphor includes key elements and systems communication, provides an upper level vision of the system. That is why some people suppose it as an architectural vision. Undoubtedly, metaphor is far away from those technical issues discussed in software architecture matters.

Paying attention on XP process and activities, an architectural movement along with architecture technics is not observed and metaphor is just a simplified picture of the system. As a result, metaphor plays the role of a product to some extent so that any technical activity has not been proposed to gain it. For instance, quality properties have special significance in the current common software architecture, while they are not mentioned in XP approaches at all.

Second method: the first step of creating a superior model, which, in fact, illustrates a superior design of the system as well. This step also covers the initial activities of the project and about 10% of the project is devoted to it. This position in the beginning of the project could give the architect and architectural activities the chance to consider architectural policies in order to develop the system. Here again, the interior design and architecture yielded through application of properties remain unanswered and might not be answered by the firs architect. This issue shows the necessity of architectural activities presence during the development, because the architecture is not accomplished and faces changes during the development and needs to be noticed and assessed. In the third approach, superior design or architecture in pre-match happens and offers a suitable opportunity for architectural activities to be emerged.

Unlike approaches such as XP, scrum approach mainly has a managerial point of view to development rather than a concentrated viewpoint on the development of the system by individuals. In this method the majority of development activities are done in the second phase which is called development phase. Other activities including assessment, design, accomplishment, testing, and delivery are performed in periods called SPRINT.

The input and driving force of these periods are the demands through which production increases. In the third phase cohesion, testing the system, documentation, and the final emission are done.

Conclusion

There is an ever growing demand for software development methodologies which are flexible toward changes and able to satisfy customer's desires in the shortest possible time. Common software development approaches are heavy and on the basis of perfect and complete planning and also bear huge documentation volume. To gain these goals, quick transferring or cheap approaches have been introduced in recent years. Agile methodologies include an extended domain of activities such as those related to collecting demands, design, performance, test, etc. Moreover, all the principles of quick transferring approaches are supported in the following way:

Emphases on the cooperation and collaboration of effective human roles in the project instead of process and tools, by means of holding regular meetings to be informed about the project status and team members performance and to recognize the remained works as well as the troubles to continue the job.

Applicable software creation in determined periods of time instead of perfect complete documents: by means of short repetitions.

Communication and cooperation between developers and customers instead of contracts: by using the customers as the team members and defining the system demands, changes requests, participation in evaluations and designs, and running acceptance tests by the customer. Agile methodologies are generalized and complete models of agile processes introducing a coherent framework for agile processes management.

The project management processes of the agile methodologies are categorized according to the proposed framework and also an agile management development framework is introduced in software creation processes.

References:

Hasani Sadi, M. and Ramsin, R., APM: A Generic Framework for Agile Project Management, Submitted to the 33th Annual IEEE Software Engineering Workshop (SEW'09).

Hasani Sadi, M. and Ramsin, R., APM3: A Methodology Metamodel for Agile Project Management, To be published in proceedings of 8th International Conference on Software Methodologies, Tools and Techniques (SoMeT'09), 2009.

Hasani Sadi, M. and Ramsin, R., FRAME: A Generic Fractal Process Metamodel for Agile Methodologies, Accepted in the 7th International Conference on Software Engineering Research, Management and Tools (SERA'09), 2009.

Lyneis, J., Ford, D., System Dynamics Applied to Project Management: A Survey, Assessment and Directions for Future Research, System Dynamics Review, vol. 23, 2007, pp. 157–189.

Project Management Institute, A Guide to the Project Management Body of Knowledge (PMBOK Guide), Fourth Edition Exposure Draft, PMI, 2008.

Ramsin, R., Paige, F., Process-Centered Review of Object-Oriented Software Development Methodologies, ACM Computing Surveys, vol. 40, No.1, 2008, pp.1–89.

Sutherland, J., Jakobsen, C. R., Johnson, K., SCRUM and CMMI Level 5: The Magic Portion for Code

Sutherland, J., Viktorov, A., Blount, J., Puntikov, N., Distributed Scrum: Agile Project Management with Outsourced Development Teams, 41st Hawaii International Conference on System Sciences(HICSS 2007), 2007, p.274.

Warriors, 42nd Hawaii International Conference on System Sciences (HICSS 2008), 2008, p. 466.