Neha Muralidharan. International Journal of Engineering Research and Applications
www.ijera.com
ISSN: 2248-9622, Vol. 10, Issue 12, (Series-IV) December 2020, pp. 01-07

RESEARCH ARTICLE OPEN ACCESS

"Is Scrum The Best Of Best?"

Neha Muralidharan
Vellore Institute of Technology, Kelambakkam - Vandalur Rd, Rajan Nagar, Chennai, Tamil Nadu 600127

ABSTRACT:
There are various methods that can be adopted for product development starting from the waterfall model to prototyping to Agile. Agile Methodology is particularly a method to effectively manage the product development that predominantly uses a combination of Iterative development and prototyping for a variety of industrial projects. This methodology is essentially used to deliver complex projects due to its ability of adaptiveness. The method focuses on collaboration (presence of various sub teams), flexibility (Due to prototyping) and continuous improvement (By adapting the iterative model) to provide high quality results. To reach its present effectiveness the model uses a variety of deliverables like product vision statement, roadmap, backlog, a release plan, sprint backlogging and regular increment, which would be thoroughly discussed in this paper. Scrum is an agile framework that is used to carry out and implement the ideas effectively provided by the agile software development. This framework was founded by Jeff Sutherland and Ken Schwaber. The reason Scrum is different from other methodologies is their roles, events and artifacts, this would also be discussed in complete detail. The real question is: Is scrum methodology the best of best? This paper aims reach the conclusion and explore other possible methods.
KEY WORDS: Agile, prototyping, Iterative, Scrum, deliverables.

Date of Submission: 14-12-2020 Date of Acceptance: 29-12-2020

I. INTRODUCTION

In the classical life cycle of the software development (Waterfall model and its variations), the development/ Progression is more linear and sequential following all the subprocess in a specific order and one acts as an input for the other, In the following order feasibility, requirement analysis, design, coding, integration and maintenance. However, this model has proved to be inefficient due to various reasons i.e. Difficulty in accommodating change (one mistake may cause huge money and resource wastage), The requirements must be stated in the beginning of the product development(initial stage) in order to avoid confusion later, Client's approval is take only at the final stage, Difficulty during integration, Not all that efficient. With the above demerits it is clear that the model does not match the reality. After several evolutions, inspections and adopted changes, one of the proven methodical model is the Agile system. The term agile itself means "the ability to move quickly and easily". This system provides justice to its name. The agile software development follows the iterative development. The method promoted project discipline along with an effective maintenance and management process where the required changes are adopted after frequent inspection. An agile software development process always starts by defining the users and the requirements in detail. A certain product owner along with a cross- disciplinary team to deliver the expected result. Now that we have a brief idea of the models in software development. Another popular process is the Scrum. Scrum is a part of the Agile methodology(A subset). This is the most widely used lightweight process framework. A scrum process can be distinguished from the agile by specific concepts and practices. Essentially the Scrum framework is said to improve the quality of deliverables, cope better with change and control the schedule and state with specific accordance. To achieve the above scrum works based on roles. A scrum team specifically requires three roles:

1. Scrum product owner: product owners are the topmost nodes of the product tree. They take responsibility of understanding the business/ market along with the client requirements and refining it along with the customer. Their main aim is to seize backlogs.
- Guides the teams in an orderly fashion.
- Builds and manages the project backlogs.
- Finalizes the end of the product and concludes when to ship it.

Note: Product owner is an individual who is not always the product manager.

2. Scrum master: the second layer of the graphs comprise of the masters who monitors an individual team unlike the owner who takes the responsibility for the entire project. An effective

scrum master is said to understand the working and the progress of the entire team in detail. Thus with this knowledge the master would be able to schedule, facilitate and set the delivery flow effectively.

3. Scrum development teams are the lower most tier of the hierarchy and these are the members who actually perform tasks for sustainable development practices. The team comprises of five to seven member. This principle is adopted from the CEO of amazon who said "If a team can't be fed by two pizzas the it is too big.".

The progress done by the teams are measured in sprints. A sprint is usually as long as a fortnight. The scrum team drives the plan for each sprint, and display the estimated amount of work that can done during that period.

Scrum is a popular agile framework which is often misunderstood with agile itself. This paper aims at providing a comparative analysis between scrum and other said profound frameworks, considering each development step. Here we will explore other top veneers like kanban, lean etc.

II. REVIEW OF LITRETURE

by Enric Senabre Hidalgo Internet Interdisciplinary Institute (Universitat Oberta de Catalunya), Av. Carl Friedrich Gauss, 5, 08860,Castelldefels, Barcelona, Spain

This paper explores the adoption of agile methods for the management of projects. The utilization of the scrum framework is currently being expanded to many other types of organizations. The study showcases the extent to which key principles and tools are usually used in scrum can contribute to the collaborative management and coordination of tasks in various research processes. According to the results of integrating agile methods and their principles for various interdisciplinary collaborations requires a very high degree of flexibility.

By Anum Ali ,Department of Computer Science Lahore College for Women University, Lahore, Pakistan ,Mariam Rehman

Agile scrum methodology has been evolved over the time largely through software industry where it has grown and developed with a great progress. The research work presented has proposed a framework by identifying critical elements of application of agile scrum methodology in software industry. The framework proposed consists of four elements, i.e. technical, people, environmental and organizational. The proposed framework is validated through statistical analysis, which comprises of The Structural Equation Modeling (SEM) data is collected from those software industry personals who are working on agile methodology. This research concludes that out 18 hypothesis 15 hypothesis were found to be significate. These significant hypothesis includes Training & Learning, Societal Culture, Communication & Negotiation, Personal Characteristics, Customer collaboration, Customer commitment, Decision Time, Team Size, Corporate Culture, Planning, Control, Development, Information Administration, and Working Environment.

By Ng, G. C. (2018). A Study of an Agile methodology with scrum approach to the Filipino company-sponsored I.T. capstone program. International Journal of Computing Sciences Research, 2(2), 68-88. doi:10.25147/ijcsr.2017.001.1.25

This research aims to show the relevance of the student projects which are sponsored by company clients in the University of Asia through the use of Agile Methodology with the help Scrum Methodology concept. The modified program is employed in two batches with content analysis and the survey results are shown as benchmarks. Various surveys at the end of the sprints for both clients as well as students showed that the length of the sprint was a critical factor in the development of the particular information system, and that the students always learned from addressing all the additional difficulties such as academic load, team pressure and communication problems. Over-all results showed that clients were impressed and very keen to adopt the student works. Maintainability aspects of the research can be analysed for future research. The increase in the sample size with more additional batched helps in leading to the discovery of various new factors that were not seen previously. The research could also help in improvement of other capstone programs and also helped in improving communication with clients.

By Bernardo Vasconcelos de Carvalho, Carlos Henrique Pereira Mello University Center of Itajubá Federal University at Itajubá

The objective of the paper was to present a detailed analysis, review and classification of the literature on the agile ,scrum methodology. Research was done to find out the benefits of using scrum methodology. Research has concluded that various literature on this topic is still scarce. Conclusion was drawn that there is demand for the generation of scientific knowledge on this subject.

By Astha Singhal B.Tech.(CSE) RKGITW Ghaziabad , Divya Gupta B.Tech.(CSE) RKGITW Ghaziabad

The most common mythology in the product development domain of the software

industry is Scrum. Scrum uses an increment and loop based approach (involving sprints), unlike the other development processes. The major concern of the clients, whose employees use scrum is addition and the involvement of the changing and increasing requirements. Scrum involves structural organizations of teams and are self-sufficient for the tenure of a single sprint. A clear review has been done on the workings and adaptations in the implementations of scrum. This paper was an add on information for the professionals interested in Scrum methodology.

III. TOP MODELS IN PRODUCT DEVELOPMENT

Scrum

Scrum works in a very efficient manner. It follows the rugby team characteristics. The rugby team characteristics is compared to the scrum methodology as all the developers put their heads together in order to discuss the difficult problems. The team that develops the software start by making a wish list. Known as the product backlog. The list contains the following:

1.The backlog.
2.Tasks that need to be completed.
3.Time taken to complete these tasks.

Scrum depends on a concept called sprints:

1.its the actual time period of the software development.
2. it maybe from a minimum of one week to one month to complete each item from the backlog.
3.every sprint ends with a review.
4. the team chooses the next backlog to develop.
5. the sprint continue till the project time is done or if the entire budget has been spent.

Lean

Lean is the process of reducing waste and adding on the customer defined values to the products and services. The meaning of lean is coined differently around the globe.

In United states: lean is a system of tools and technique used to reduce waste and to add value to every process. In Japan: in japan its considered as a mindset and not the set of tools.

The basic principles of the lean methodology is to eliminate all forms of waste and to increase the customer perceived value with everything that's done. Jim Benson of Modus Cooperandi defined Lean methodology as: "Lean is both a philosophy and a discipline which, at its core, increases access to information to ensure responsible decision making in the service of creating customer value."In short lean means to do more with less while doing it better.

Kanban

It is a way of managing the continual growth in the product development process with a certain emphasis in a way to not overburden the team. Kanban is a process similar to scrum whose entire development process depends on the team. Kanban methodology revolves around three principles:

1.workflow: It visualizes exactly what the team does on that day. It helps to plan it out as its easier and very informative.
2.work in progress(wip): it helps manage the "flow-based approach" in order to make sure the teams start any work and not take up too much work at one go.
3.enhancing the work flow: when a particular work is completed, the next highest work of priority is pulled from the backlog into play. Kanban ensures the promotion of continuous collaboration and encourages learning by improving a best team workflow.

The main goal is to stop starting and start finishing.

A few benefits:

1.response to change is high
2.shorter cycle time helps deliver features faster
3.when priorities change Kanban is an ideal methodology.

Feature Driven Development

This a process that focuses on client, architecture and the pragmatic software process. Client in a feature driven development refers to the project stakeholders as in the agile methodology.

It allows to update the project frequently and recognize the errors quickly. The clients are provided information and the results at any point of time.

Basic activities:

1. To develop the model
2. To develop a feature list
3. To plan according to the feature list
4. To design according to the feature
5. To build the features

an overall model shape is created within the first two steps. The last three steps are repeated for each feature. The majority of the work in feature driven development is done in the last two steps those are the design according to the feature ad build the feature.

IV. COMPARITIVE ANALYSIS

Scrum:

A scrum is a frame work that significantly empowers agile. Although scrum has its own benefits and advantages, it is difficult to practice and actualize. Scrum allows the implementation of diverse techniques and procedures with the help of a cross functional team. In scrum there is self-

association of groups that are not repelled by any outside individuals. Choosing a team in scrum may be quite challenging as the group must be small enough to handle change and sufficiently big to manage efficient work. In a single iteration scrum has to focus on planning just enough, building with minimum set of features and finally reviewing the product. The model promises to shup valuable increment of work at the end of each sprint. It addresses complex problems (hence require to be adaptive) while efficiently delivering products of the highest value. It implements a process where progress is based on observations of reality, not fictious plans. Overall, Scrum is a very adaptive framework.

Kanban:
Kanban is a development model that gradually improves tries and tested products.

Current backlogs	To do	In progress	Work Completed

Kanban visualizes manageable chunks of work using the above table as it progresses through the flow. The model is designed to meet minimal resistance while allowing continuous incrementing small changes. Kanban helps to analyze complex attributes like risks associated to complete the most work on time. Overall, Kanban is for a product that works fine but still needs some optimizations.

Scrum and Kanban allow for complex adaptive tasks to be reduced down and be completed effectively. Both involve continual improvement, process optimization, and they share a similar focus on highly visible workflow.

Differentiating between Kanban and Scrum sets down different paths. Kanban limits work whereas scrum limits time. Scrum primarily focusses on scheduling, weighs the story points and decides how many of those must be completed in one sprint providing a shippable product after every iteration. On the other hand. Kanban has no time boxes and the improvements are expected in an evolutionary fashion.

Kanban and scrum are different from each other in different aspects. Firstly, Roles and responsibilities: Every individual in scrum has a fixed role i.e product owner, scrum master etc. despite the development team being role-fluid, but Kanban allows flexibility in terms of individuals. Secondly, Team and commitment: every individual in scrum must commit to doing a specific amount of work in Kanban, commitment is optional for teams. Thirdly, Obstacles are immediately dealt with in scrum, whereas, they are left to be dealt with later in Kanban. Fourthly, Types of teams: in scrum it is necessary to create cross-functional teams. Kanban usually has specialized teams throughout the course of the project. Lastly, Objective of the team: Scrum's main objective is team collaboration to complete the task, whereas, Kanban usually strives to achieve goals.

Most companies have adopted either scrum or Kanban for their product development. Teams in companies like Apple, Google, Amazon predominantly adopted scrum whereas some like Hp, Pixar, Spotify have preferred Kanban. When choosing between Kanban or scrum, the individuals distinction does not always have to be made because Kanban and scrum cam go hand-in-hand (Scrumban).

Lean:

Expanding about lean, some may find agile and lean contradictory in nature but in fact they are quite complementary. Lean is sometimes categorized as another agile method rather they are different and both work in new ways to bring better results. The lean mindset encourages to take system view of value stream, this involves considering beyond the software team, considering the entire value stream from concept to money, similar to Kanban. The model majorly focusses on improving the flow of value. Lean principles identify ways to eliminate wastes like task switching, hand offs, partially done work and extra features etc. but the Scrum and Kanban resolve this aspect in the phase of requirement analysis itself. While agile principles teach us to work in short cycles and deliver more valuable product faster, lean principles point to the limiting of queues which prevents work from piling up and invites the notion of pull: "only pull work when you have the space to invite more work in the process". Lean and agile are fairly common, both encourage to improve quality, amplify learning, continuously improve and empower people. Overall, both lean and agile are aimed at achieving a successful business product. However, they serve different purposes and tasks. Lean is a wider concept than agile as it is a smart approach that handles all losses from money to labour to energy.

Feature driven development

In the Feature Driven Development model, the features are the corner stones of the product development. The requirements are introduced as the features and are increased at every stage of the development unlike in scrum where it has to be defined initially. This is implemented best when the internal teams are large. This model also adopts

timely increments. This approach can be considered an intermediate between waterfall model and the agile software development technologies (Scrum). Feature Driven Development is predominantly chosen for large scale product development and allows adaptability. Its key principles focus on well-documented software development practices. The models hold a few consequences and thus not voted as the best such as: cannot be used for small scale projects with more firm deadlines, the success of this model completely depends on having a highly-skilled team leads monitoring the process throughout the development cycles(similar to the scrum masters in the scrum development) and the documentation must be well maintained.

V. IMPLICATIONS

Reduced Ambiguity

One of the most common issue in the development process is the increased threat of uncertainty and ambiguity. This issue already poses a severe threat in most product development methods, but this is even higher at firms/ methods where informal communications happen in individual/distributed teams. The issues can be caused at any stage of the development process i.e administrative procedures, management, requirements and even in the design stage. Depending upon the team experience and the possessed domain knowledge the uncertainty may improve or worsen. However scrum intelligently avoids this process, with inherent availability of the team communications and coordination. On following the regularized sprits combined with a daily meeting this issue is expected to reduce greatly.

Maximized stability

Another important feature in product development is the stability. Scrum allows this easily as these practices in distributed environment (with effective communication) maximizes stability of the development process.

Understanding deficiencies

Dependencies are burners in every environment. These need to be effectively managed by ensuring the optimal frequency, size and coordination type among the contributors. Dependencies and deficiencies play a major role in the planning and the implementation of the project (Allows to have a firm grip on the overall synchronization).

Facilitate coordination

As mentioned in the first implication communication and coordination help to resolve a great deal of problems in a certain work/model space. Especially in distributed teams, coordination is one way to not that the process is in well shape. This feature can be bought together in processes, structural design, management practices. These choices can be very useful in risk management. These methods are carefully utilized in scrum methodology at its best. An evolved version of scrum is the scrumban. Scrumban brings an up-to-date task backlog. Thus allowing the team to visualize their improvements. Depending on the development pace the meetings can be conducted accordingly to ensure the sync.

VI. CONCLUSION

In this paper, we have carefully analyzed the differences between four popular product development models: lean, Kanban, scrum and feature driven development. Analyzing these frameworks can be like contrasting oranges with apples, because some of these methods may piggy-back on or complement each other, particularly when they are implemented in various parts of the development lifecycle. It is not always practicable to stick on to a specific practice to just one framework, and teams often use hybrid methodologies to organize their work and development. However, the most preferred method would be the Scrumban. As the name suggests Scrumban is a hybrid or a third way of Scrum and Kanban. This method tries to bring out the best characters from each. Scrum allows us to complete a fixed requirement with a specific deadline. On the other hand, Kanban allows to give flexibility to the team as to schedule their work within that time frame(limiting the time for WIP). The most generic scrumban board holds the following swim lanes: Backlog, On Deck, Next, Specify, Specified, Doing, Pending Testing, Testing, Deployment, Done. Srumban was initially developed for IT based projects, the principles are generic and can be incorporated into any project.

REFERENCES

[1]. Agile Alliance, "Agile Manifesto," Online at <http://www. agilemanifesto.org>, vol. 6, 2001.
[2]. S. W. Ambler, “The Non-Existent Software Crisis: Debunking the Chaos Report,” Available: <http://www.drdobbs.com/architecture-and-design/the-non-existent-software-crisis-debunki/240165910>,11 September, 2014.
[3]. N. Nikitina and M. Kajko-Mattsson, "Developer-driven big-bang process transition from SCRUM to Kanban," in Proceedings of the 2011 international conference on software and systems process, 2011, pp. 159-168.
[4]. H. Kniberg and M. Skarin, Kanban and SCRUM-making the most of both: Lulu. com, 2010.

[5]. R. Cuellar, "Kanban for Help Desks: Managing the Unplannable," Cutter IT Journal, vol. 24, p. 23, 2011.
[6]. R.H. Al-Ta'ani and R. Skarin, "A Framework for Requirements Prioritisation Process in an Agile Software Development Environment: Empirical Study," International Journal on Advanced Science, Engineering and Information Technology, vol. 6, pp. 846- 856, 2016.
[7]. A. Shalloway, "Demystifying Kanban," Cutter IT Journal, vol. 24, p. 12, 2011.Agile Alliance, "Agile Manifesto," Online at <http://www. agilemanifesto.org>, vol. 6, 2001.
[8]. S. W. Ambler, "The Non-Existent Software Crisis: Debunking the Chaos Report," Available: <http://www.drdobbs.com/architecture-and-design/the-non-existent-software-crisis-debunki/240165910>, 11 September, 2014.
[9]. N. Nikitina and M. Kajko-Mattsson, "Developer-driven big-bang process transition from SCRUM to Kanban," in Proceedings of the 2011 international conference on software and systems process, 2011, pp. 159-168.
[10]. H. Kniberg and M. Skarin, Kanban and SCRUM-making the most of both: Lulu. com, 2010.
[11]. R. Cuellar, "Kanban for Help Desks: Managing the Unplannable," Cutter IT Journal, vol. 24, p. 23, 2011.
[12]. R.H. Al-Ta'ani and R. Skarin, "A Framework for Requirements Prioritisation Process in an Agile Software Development Environment: Empirical Study," International Journal on Advanced Science, Engineering and Information Technology, vol. 6, pp. 846- 856, 2016.
[13]. A. Shalloway, "Demystifying Kanban," Cutter IT Journal, vol. 24, p. 12, 2011.
[14]. Agile Alliance, "Agile Manifesto," Online at <http://www. agilemanifesto.org>, vol. 6, 2001.
[15]. S. W. Ambler, "The Non-Existent Software Crisis: Debunking the Chaos Report," Available: <http://www.drdobbs.com/architecture-and-design/the-non-existent-software-crisis-debunki/240165910>, 11 September, 2014.
[16]. N. Nikitina and M. Kajko-Mattsson, "Developer-driven big-bang process transition from SCRUM to Kanban," in Proceedings of the 2011 international conference on software and systems process, 2011, pp. 159-168.
[17]. H. Kniberg and M. Skarin, Kanban and SCRUM-making the most of both: Lulu. com, 2010.
[18]. R. Cuellar, "Kanban for Help Desks: Managing the Unplannable," Cutter IT Journal, vol. 24, p. 23, 2011.
[19]. R.H. Al-Ta'ani and R. Skarin, "A Framework for Requirements Prioritisation Process in an Agile Software Development Environment: Empirical Study," International Journal on Advanced Science, Engineering and Information Technology, vol. 6, pp. 846- 856, 2016.
[20]. A. Shalloway, "Demystifying Kanban," Cutter IT Journal, vol. 24, p. 12, 2011.
[21]. Agile Alliance, "Agile Manifesto," Online at <http://www. agilemanifesto.org>, vol. 6, 2001.
[22]. S. W. Ambler, "The Non-Existent Software Crisis: Debunking the Chaos Report," Available: <http://www.drdobbs.com/architecture-and-design/the-non-existent-software-crisis-debunki/240165910>, 11 September, 2014.
[23]. N. Nikitina and M. Kajko-Mattsson, "Developer-driven big-bang process transition from SCRUM to Kanban," in Proceedings of the 2011 international conference on software and systems process, 2011, pp. 159-168.
[24]. H. Kniberg and M. Skarin, Kanban and SCRUM-making the most of both: Lulu. com, 2010.
[25]. R. Cuellar, "Kanban for Help Desks: Managing the Unplannable," Cutter IT Journal, vol. 24, p. 23, 2011.
[26]. R.H. Al-Ta'ani and R. Skarin, "A Framework for Requirements Prioritisation Process in an Agile Software Development Environment: Empirical Study," International Journal on Advanced Science, Engineering and Information Technology, vol. 6, pp. 846- 856, 2016.
[27]. A. Shalloway, "Demystifying Kanban," Cutter IT Journal, vol. 24, p. 12, 2011.
[28]. Agile Alliance, "Agile Manifesto," Online at <http://www. agilemanifesto.org>, vol. 6, 2001.
[29]. S. W. Ambler, "The Non-Existent Software Crisis: Debunking the Chaos Report," Available: <http://www.drdobbs.com/architecture-and-design/the-non-existent-software-crisis-debunki/240165910>, 11 September, 2014.
[30]. N. Nikitina and M. Kajko-Mattsson, "Developer-driven big-bang process transition from SCRUM to Kanban," in Proceedings of the 2011 international conference on software and systems process, 2011, pp. 159-168.
[31]. H. Kniberg and M. Skarin, Kanban and SCRUM-making the most of both: Lulu. com, 2010.
[32]. R. Cuellar, "Kanban for Help Desks: Managing the Unplannable," Cutter IT Journal, vol. 24, p. 23, 2011.
[33]. R.H. Al-Ta'ani and R. Skarin, "A Framework for Requirements Prioritisation Process in an Agile Software Development Environment: Empirical Study," International Journal on Advanced Science, Engineering and Information Technology, vol. 6, pp. 846- 856, 2016.
[34]. A. Shalloway, "Demystifying Kanban," Cutter IT Journal, vol. 24, p. 12, 2011.

[35]. Agile Alliance, "Agile Manifesto," Online at <http://www. agilemanifesto.org>, vol. 6, 2001.
[36]. S. W. Ambler, "The Non-Existent Software Crisis: Debunking the Chaos Report," Available: <http://www.drdobbs.com/architecture-and-design/the-non-existent-software-crisis-debunki/240165910>, 11 September, 2014.
[37]. N. Nikitina and M. Kajko-Mattsson, "Developer-driven big-bang process transition from SCRUM to Kanban," in Proceedings of the 2011 international conference on software and systems process, 2011, pp. 159-168.
[38]. H. Kniberg and M. Skarin, Kanban and SCRUM-making the most of both: Lulu. com, 2010.
[39]. R. Cuellar, "Kanban for Help Desks: Managing the Unplannable," Cutter IT Journal, vol. 24, p. 23, 2011.
[40]. R.H. Al-Ta'ani and R. Skarin, "A Framework for Requirements Prioritisation Process in an Agile Software Development Environment: Empirical Study," International Journal on Advanced Science, Engineering and Information Technology, vol. 6, pp. 846- 856, 2016.
[41]. A. Shalloway, "Demystifying Kanban," Cutter IT Journal, vol. 24, p. 12, 2011.
[42]. Agile Alliance, "Agile Manifesto," Online at <http://www. agilemanifesto.org>, vol. 6, 2001.
[43]. S. W. Ambler, "The Non-Existent Software Crisis: Debunking the Chaos Report," Available: <http://www.drdobbs.com/architecture-and-design/the-non-existent-software-crisis-debunki/240165910>, 11 September, 2014.
[44]. N. Nikitina and M. Kajko-Mattsson, "Developer-driven big-bang process transition from SCRUM to Kanban," in Proceedings of the 2011 international conference on software and systems process, 2011, pp. 159-168.
[45]. H. Kniberg and M. Skarin, Kanban and SCRUM-making the most of both: Lulu. com, 2010.
[46]. R. Cuellar, "Kanban for Help Desks: Managing the Unplannable," Cutter IT Journal, vol. 24, p. 23, 2011.
[47]. R.H. Al-Ta'ani and R. Skarin, "A Framework for Requirements Prioritisation Process in an Agile Software Development Environment: Empirical Study," International Journal on Advanced Science, Engineering and Information Technology, vol. 6, pp. 846- 856, 2016.
[48]. A. Shalloway, "Demystifying Kanban," Cutter IT Journal, vol. 24, p. 12, 2011.
[49]. Agile Alliance, "Agile Manifesto," Online at , vol. 6, 2001.
[50]. S. W. Ambler, "The Non-Existent Software Crisis: Debunking the Chaos Report," Available: , 11 September, 2014.
[51]. N. Nikitina and M. Kajko-Mattsson, "Developer-driven big-bang process transition from SCRUM to Kanban," in Proceedings of the 2011 international conference on software and systems process, 2011, pp. 159-168.
[52]. O. N. Al-Allaf, "The Adoption of Agile Processes in Large Web Development Enterprises: A Survey in Jordan," IJWA, vol. 2, pp. 206-216, 2010.
[53]. F. Ahmad, F. Baharom, and M. Husni, "Agile development methods for developing web application in small software firms," in Knowledge Management International Conference (KMICe), 2012.
[54]. S. Denning, "Why Agile can be a game changer for managing continuous innovation in many industries," Strategy & Leadership, vol. 41, 2013, pp. 5-11.
[55]. Tolfo, C., & Wazlawick, R. S. (2008). The influence of organizational culture on the adoption of extreme programming. Journal of Systems and Software, 81(11), 1955–1967. doi:10.1016/j.jss.2008.01.014
[56]. F. Kanwal, K. Junaid, and M.A. Fahiem, "A hybrid software architecture evaluation method for fdd-an agile process model," in Computational Intelligence and Software Engineering (CiSE), 2010 International Conference on, pp. 1-5, 2010.