Volume 5, Issue 7, July 2015 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Understanding Agile System Development Methodologies

Tefo Sekgweleo*
Polytechnic of Namibia,
Namibia

Abstract*— *Software development methodology plays a vital role in systems development life cycle. It is a framework that guides the systems development team in achieving what the customer/user has requested. Decision making may impact the systems development team positively or negatively. Hence, understanding strengths, limitations, how, why and who can use the software methodology is imperative. It helps all the stakeholders involved in a systems development team to make informed decision for a particular project. Hence, the software development methodology is not a silver bullet for all the projects.

Keywords— *Systems Development Life Cycle (SDLC), Information Technology (IT), Systems Development, Agile*

I. INTRODUCTION

There is a stage in life where a need arises within the organisation to purchase or develop a system. This need may require the organisation to undergo through a system development life cycle (SDLC) in order for it to increase productivity to remain competitive. To achieve that, most organisations have to make a decision to developing a system or purchase a system. Therefore, there are various systems development methodologies that can be followed after making that crucial decision because there are finances involved. [1] Define SDLC as a guideline and logical process used by system developers to develop systems. According to [2], SDLC stipulates the required ways that comprises various stages and activities to successfully develop the system. However, it should be taken into consideration that the methodology is not one size fit all. The software development team has to carefully select the appropriate methodology for a particular project they are undertaking.

These methodologies serve as a framework to be followed by a software development team. It can also be used to ensure that the designed solution meet the user requirements that supports business strategic goals and objectives. The SDLC can be either agile or traditional. However, both methodologies are made up of various stages namely analysis, design, development, implementation and maintenance [3]. However, the main purpose of this paper is to examine the agile methodologies and to understand why, who and how they are used and also highlight the limitations thereof.

II. AGILE SYSTEMS DEVELOPMENT

Nowadays, the business environment is so dynamic and requires organisations to constantly change its requirements to adapt the new ways of doing business. These organisations need systems that will enable them to compete with their rivals. The continuous requirement change has been a big issue within systems development arena. However, the introduction of agile addressed that issue and others such as user involvement, rapid and flexible response to change and frequent delivery [4]. Agile systems development is described as a group of systems development methodologies aiming at quickening the development processes for responding to continuous requirements change [5]**.** It is impossible for the user/customer to know all the system requirements prior to the development of the system. Therefore agile methodologies make it possible address that problem.

[6] defines agile methodology as "subset of iterative and evolutionary methods that are based on iterative enhancement and opportunistic development processes". In agile, a system is developed in cycles also known as iterations. Within the iteration various systems development activities occur. [7] posits that, at the end of each iteration, the system is released to the customer for use. [6] stipulates that every iteration leads to an iteration release that integrates the entire system across the team and is a growing and evolving subset of the final system. The use of the system, enable the customer to figure out what they really need. Therefore, the agile process allows rapid response to changes in requirements, detailed and perpetual collaboration between the development team and the customer [8]. Team work is what brings the two together to achieve the desired solution.

Agile methodologies are based on iterative and incremental development whereby requirements and solutions keep on evolving. It promotes user involvement, development, teamwork, collaboration and process adaptability throughout the life cycle of the systems development project. The agile systems development methodologies include eXtreme Programming (XP), Scrum, Dynamic Systems Development Method (DSDM) and Lean Software Development [9]. The methodology is chosen based on the scope of the software development project or according to the organisation standard. The agile methodologies are briefly explained below.

III. EXTREME PROGRAMMING

Rapid changing requirements in systems development became a stumbling block in the success of implementing systems. As a result, eXtreme Programming (XP) was introduced by Beck in 1999 to help eradicate or solve the mentioned problem [10]. XP was intended to improve the responsiveness towards the rapid changing customer requirements and the quality of the systems. [11] posits that XP relies too much on constant communication between stakeholders which promotes firm feedback and good response to any changes that might be required. It is also concerned about the actual programming practices because it is low risk, efficient and flexible development approach that is suitable for rapid changes to the requirements [12].

XP is made up of five stages namely exploration, planning, iterations to release, productionising, maintenance and death [13]. The figure below demonstrates the five stages of XP life cycle.

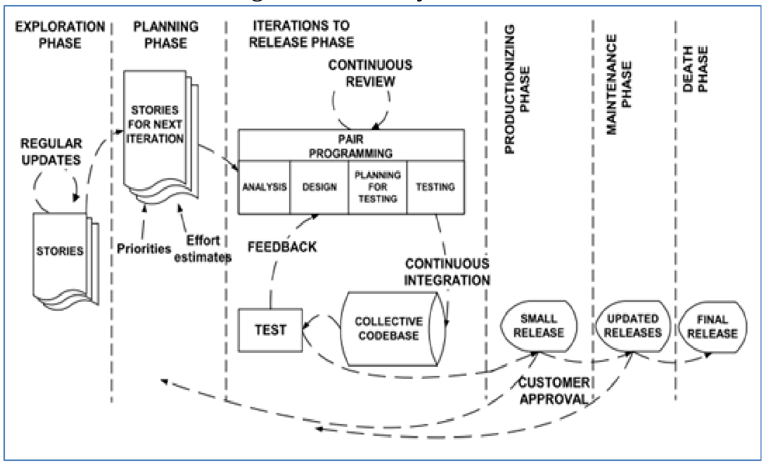

Fig. 1 [14]

Prior to the commencement of the systems development it is imperative to form a team that will sit together to ensure that the requested system is delivered. [10] argue that XP focuses on teamwork and skilled systems developers to confidently react to changing customer requirements anytime in the life cycle. XP is defined by [15] as a process that enables the co-ordination of team activities as well as aiding in the rapid software development of projects to satisfy customer's needs. The core values described below play a vital role in bringing the entire team together with its simple practices.

Table 1: XP Core Values [16]

XP Core Values	Description
Communication	Effective communication between project stakeholders helps resolve issues as they arise
Simple	XP emphasises starting with a simple solution and adding extra functionality later
Feedback	Testing the solution earlier enables the developers to get continuous feedback which enable them to resolve errors in the early iterations
Courage	Developers need to have courage for fixing errors, re-writing or removing the completed tasks caused by changing requirements

The above mentioned core values together with the twelve best practices described below best defines extreme programming. XP emphasises team work. As a result all the members of the team are equal partners in a collaborative team.

Table 2: XP Best Practices [17]

XP Best Practices	Description
Planning game	It involves the efforts required to decide on implementing the customer stories
Small releases	The system is developed in a series of small releases that are continuously updated
Metaphor	Pictorial presentation of how the system works enable the team members to better understand the system
Simple design	Keep the system design as simple as possible to accommodate changing requirements
Testing	Conduct testing in order to build high quality system
Refactoring	The code is rearranged to remove duplication, improve communication and simplify the code without affecting the functionality of the system
Pair programming	The code is written by two developers on a single computer
Collective code ownership	The code belongs to the development team and can be updated by any developer when changes are required
Continuous integration	Any new code is thoroughly tested and integrated to the system

40-hour week	A maximum of 40 hours per developer to work is permissible
On-site customer	The customer's availability is required at all times to respond to the development team's questions
Coding standards	Coding rules that exists among the team enables the developers to follow the same pattern of coding which leads to common understanding

The XP teams work in a series of short development cycles, which is intended to improve productivity and introduce checkpoints where new customer requirements can be adopted short development cycles. At the beginning of each iteration, the team gets together with the customer for a planning meeting. In that meeting, they go over the features the customer requires in that iteration and each feature is broken down into individual engineering tasks. During the rest of the iteration, the team implements the features they agreed upon. At the end of the iteration the programmers delivers a working system to the customer. XP is severely dependent on continuous communication between stakeholders, tight feedback loops to clarify, specify feature implementation and to respond to change [11]. Below is the XP project development model.

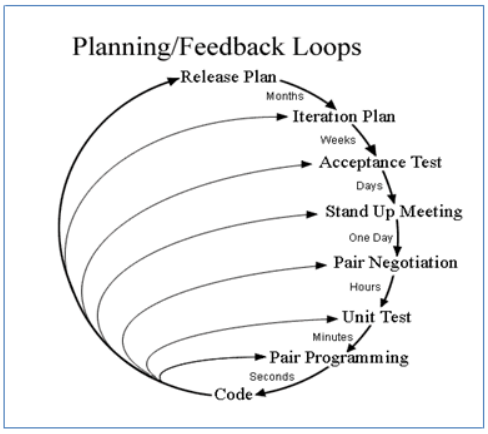

Fig. 2 [12]

However, the potential weaknesses of XP include problems with unstable requirements, lack of documentation design and XP does not provide data gathering guidance (Khan et al., 2011).

IV. SCRUM

Scrum is one of the several agile light-weight methods that use an iterative and incremental approach for the systems development. The scrum methodology is based on the game of rugby and was introduced by Schwaber in 1995 [18]. The purpose of scrum in the field of play is to resume the play quickly, safely and fairly after stoppage (Gang et al., 2011). [19] posit that Scrum methodology inspires independence among team members, provides satisfaction among employees and enhances team spirit which increases the quality of work. Therefore, quality and the best results can be achieved through the team that is prepared to function as a unit to meet the organisational goals and objectives.

The iteration in scrum is referred to as a sprint. In scrum, work is divided into a set of features known as sprints which typically lasts a month in duration [20]. The scrum method brings a small team together to work on those specified features. The intension of this team is not only to reveal organisational problems but to solve them through hard work with open minds and to also to demand changes within the organisation [21]. The Scrum method is made up of three stages which are divided into sub stages. [19] states the stages as follows: the initial stage is pre-game (sub-processes: planning and system architecture) followed by the game (sub processes: develop, wrap, review and adjust) and post-game (closure). Below is the scrum development life cycle:

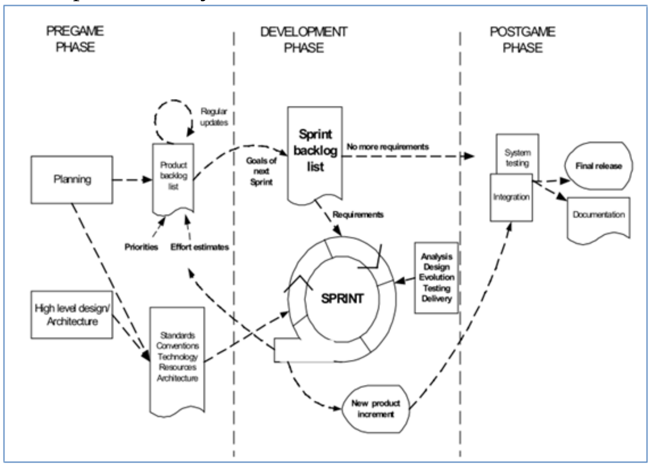

Fig. 3 [14]

Scrum is ideally suited for projects with rapidly changing or highly evolving requirements. It is through hard work, creativity and the willingness to work as a team that best results are achieved. The vision of the development team is to deliver a final working product which is according to the changing requirements [22]. At the closure of the systems development the team is crowned the winners of the game. Hence, the organisation has achieved quality results (requested product) that fulfil the organisational requirement.

V. DYNAMIC SYSTEM DEVELOPMENT METHOD (DSDM)

The people skills together with the knowledge of software development tools and techniques play a vital role in dynamic systems development method (DSDM). In order to achieve tight delivery project timelines, DSDM combines the actual use of people's knowledge with tools and techniques such as prototyping [23]. Prototyping in DSDM is used as a mechanism that paints a clear picture of all the aspects of the system to the various stakeholders. As the other methodologies, DSDM is divided into three stages namely pre project, project life-cycle and post project [24]. These stages are also accompanied by nine principles of DSDM. They include user involvement, project team empowerment, frequent delivery, business needs are important for delivery, iterative and incremental development, changes are reversible, requirements are base lined at a high level, testing is integrated throughout the life cycle, collaboration and cooperation is vital [23].

The main focus of dynamic systems development method (DSDM) is to deliver working business solutions within tight time schedules. However, achieving the results on time requires cooperation and collaboration from all the project stakeholders. Therefore the use of DSDM enables the team to define the delivery time and to limit the resources in order to create the process that entirely satisfies the user's requirements [25]. Within DSDM, the time and resources are fixed for the life of the project which allows the requirements to be satisfied to change [26]. The customer's satisfaction becomes the priority in DSDM because at the end of the project the desired solution has to be delivered.

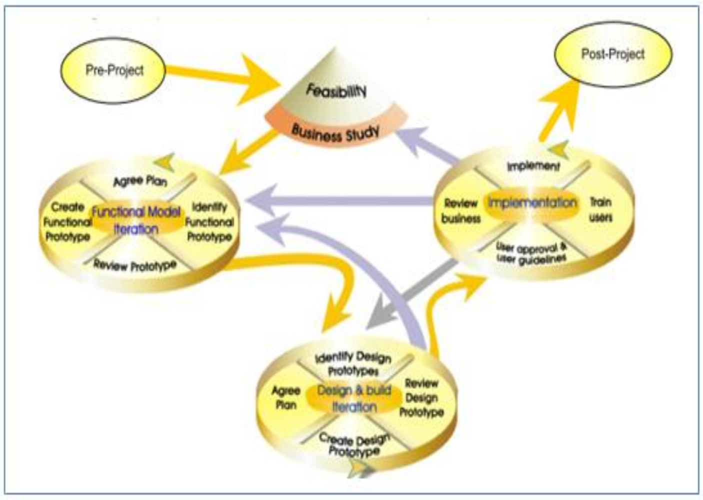

Fig. 4 [27]

VI. LEAN SOFTWARE DEVELOPMENT

Lean Software Development is actually a set of principles transferred to the software domain from manufacturing. Essentially, it is a systematic approach to identify and eliminate waste through continuous improvement. In lean the production occurs in smaller batches of high quality since the feedback loop is short and it allows controls to be adjusted more frequently and enable resources to be used efficiently without any queuing [28]. Splitting and performing tasks into smaller chunks simplifies the work and makes it more manageable.

In the automotive industry lean is about defining value in the eyes of the customer which equates to our highest priority is to satisfy the customer and deliver working software frequently in the software industry [29]. This is the principle which enables lean methodology to be adopted in both industries (auto mobile and software development). Lean software development methodology consists of seven principles: eliminate waste, amplify learning, decide as late as possible, deliver as fast as possible, empower the team, build integrity, and see the whole [24]. As stated by [30], these principles serve as a guideline to companies with regards to standardising methods, activities and work products. Standards and methods introduce uniformity and the way of working within the organisation. It makes it easy for the colleagues to take-over from another colleague who might not be present at work.

VII. WHY IS AGILE USED

Agile emphasises delivering early and often, enabling the business to begin generating a return on investment much earlier. Agile methods are intended to the dynamically changing needs of the business [31]. [8] argue that agile promotes quick response to requirement changes as well as broad and continuing collaboration between the systems development team and the customer. In return, the business gets the highest-valued features delivered early and which is of high quality. [20] argue that agile enable developers to build solutions more quickly and efficiently. There is a demand for fast delivery of software products as well as accepting changing requirements. [10] argue that agile methodologies offer organisations with the capability to swiftly advance systems development and to respond to rapidly changing requirements. Some of the principles of agile is to welcome changes to requirement late in the development process, deliver working system frequently in short space of time, satisfy the customer, enabling close relationship between business people and system developers, promoting face-to-face conversation amongst team members, simplicity and promoting sustainable development [32].

The agile methodology ensures that the time spent between the idea and the delivery has a short feedback cycles to enable any corrections to be made early. Its practices are based on program design, coding and testing that are assumed to enhance the flexibility and productivity of system development [33]. Such practices are known as iterations/cycles. With short feedback cycles quick and early adaptations to business changes is achieved.

VIII. HOW IS AGILE USED

Agile assumes that the requirements will evolve and change as the customer begins to visualize their own requirements. According to [34], agile tolerate the development team to focus on the software itself rather than on its design and documentation. The flexible nature of agile allows the system development team and customers to work with what they which will evolve until the desired solution is delivered. Therefore feedback from the customer serves as a guiding principle and clearly provides opportunities to gather missed requirements [35]. Regardless of not knowing the requirements completely in advance, the agile process allows the system development to be broken down into several iterations.

Agile focuses primarily on short iterative cycles and depend on on the knowledge within a team [20]. The team make use of the information they get from the customer and it does not matter how little it is, the development carries on. Once the first cycle is complete the system is rolled out to customer for use. Agile puts extreme emphasis on delivering working code or system to the customer as quick as possible [10]. Through the use of these short cycles the customer is able to realise the requirements they have missed or change some that they requested. [36] posit that all agile methods encourage frequent repositioning of development goals with the customers' needs and expectations in mind. It is normal for the customer to change their mind and for the system development team to readjust the program to meet the customers' needs.

IX. WHO USES AGILE

Depending on the organisational structure, the agile systems development team may comprise of the project manager, system architects, business analysts, business people, systems analysts, systems developers, systems testers and customers (end users). The roles and responsibilities differ according to the agile methodology selected but the above mentioned roles features in most agile methodologies. [38] posit that stakeholders with diverse backgrounds and goals are involved in making collaborative decisions which is the ultimate characteristic of agile development. [39] posit that the roles in Scrum include the product owner (person from business), Scrum team (developers, testers and other roles) and Scrum Master (team leader).

X. LIMITATIONS AND CHALLENGES OF AGILE

Agile methodologies were introduced to eradicate and simplify the difficulties encountered through the use of traditional methodologies. However, any new technology that is introduced comes with its strengths and limitations. Therefore, agile offers limited support to distributed development environment though it encourages face-to-face communication between developers and customers [40]. There can be other means of communication but what makes the agile team productive is the face-to-face communication. Agile also offers limited support for development involving large teams because it forces team members to be located in different locations [41]. This also defeats the purpose of face-to-face communication. Pair programming also becomes a challenge in the distributed environment because two team members has to work side by side on the same code which is another typical feature of agile practices [5]. The tendency Extreme Programming which focuses on building software products that solve a specific problem limits agile from producing reusable artifacts [42]. The best way of working is to produce the solution or components that are re-usable and capable of solving more than one problem.

Explicit knowledge (knowledge that resides within the organisation through documentation) is crucial to all organisations since it keeps the knowledge within organisation. However, agile development is dependent on tacit knowledge of the development team (knowledge that resides in the heads of people) and this can shift the balance of power from management to the development teams [38]. Therefore, this empowers the development team to dictate the terms of working.

XP cannot be utilised for medium and large project due to its poor architectural planning, over concentrating on early results, weak documentation and low levels of test coverage [43]. Agile methodologies do not allow systems be developed for reusable purposes since the focus is on that solving specific problems and not the general ones [44]. In agile, interaction and communication between the developers and customers is very crucial. The success of the project depends on those factors. [45] posit that the developer with poor social skill may have difficulties in attaining proper information with regards to a particular module in progress, as result that makes customers to be unable to provide accurate requirement for the following iteration.

XI. CONCLUSION

Prior to systems development it is vital for the systems development team to have a good understanding of what is required by the customer/user. It is crucial for the systems development team to also understand the type of project they are faced with before they could start working on it. That enables the team to decide on which system development methodology to follow. The understanding of how each methodology is applied by who, how and why is applied helps the system development team to make informed decision. The intention of most organisations when it comes to systems development is to be successful since IT enables them to be efficient and remain competitive to its counterparts. Failure is not an option to organisations and success is what is expected from the systems development team.

REFERENCES

[1] A. C. Nelson and J.T.C. Teng. "Do systems development methodologies and CASE tools decrease stress among systems analysts?" *Behaviour & Information Technology*, 19(4):307-313, 2000.

[2] M. A. Rob. "Dilemma between the Structured and Object-Oriented Approaches to System Analysis and Design." *Journal of Computer Information Systems*. 1(1):275-280, 2004.

[3] D.E. Avison and G. Fitzgerald. "Where Now for Development Methodologies?" *Communication of the ACM*, 46(1):79-82, 2003.

[4] S. Balaji and M.S. Murugaiyan. "Waterfall vs V-Model vs Agile: A Comparative Study on SDLC." *International Journal of Information Technology and Business Management*, 2(1):26-30, 2012 .

[5] S.V. Shrivastava and H. Date. "Distributed Agile Software Development: A Review." *Journal of Computer Science and Engineering*, 1(1):10-17, 2010.

[6] L. Williams. Testing Overview and Black-Box Testing Techniques. 35-59, 2006.

[7] M. Marchesi and K. Mannaro. (2008). *Adopting Agile Methodologies in Distributed Software Development*. Available at: http://veprints.unica.it/9BB5440F-68F5-49F8-A948-82DBC096E8DC/FinalDownload/DownloadId-EEFEC4C454B6189D35336BF352F28019/9BB5440F-68F5-49F8-A948-82DBC096E8DC/53/1/mannaro_katiuscia.pdf . [Accessed June,12, 2015]2008.

[8] E. Germain and P. N. Robillard. "Engineering-based processes and agile methodologies for software development: a comparative case study." *The Journal of Systems and Software*, 75:17–27, 2005.

[9] G. Lee and W. Xia. "Toward Agile: An Integrated Analysis of Quantitative and Qualitative Field Data on Software Development Agility." *MIS Quarterly*, 34(1):87-114, 2010.

[10] P. Meso and R. Jain. "Agile Software Development: Adaptive Systems Principles and Best Practices." *Information Systems Management*. 23(3):19-30, 2006.

[11] L. Layman, L. Williams, D. Damian, and H. Bures. "Essential communication practices for Extreme Programming in a global software development team." *Information and Software Technology*, 48:781–794, 2006.

[12] L. Liu and Y. Lu. "Application of Agile Method in the Enterprise Website Backstage Management System Practices for Extreme Programming." *IEEE- Consumer Electronics, Communications and Networks (CECNet), 2012 2nd International Conference on* 2012, 2412-2415.

[13] S. J. Hashmi & J. Baik. 2007. Software Quality Assurance in XP and Spiral - A Comparative Study. *5th International Conference on Computational Science and Applications*.

[14] P. Abrahamsson, O. Salo, J. Ronkainen and J. Warsta. Agile Software Development Methods Review and Analysis. 1-107. 2002.

[15] R. Agarwal and D. Umphress. A Flexible Model for Simulation of Software Development Process. *Proceedings of the 48th Annual Southeast Regional Conference.* 2010.

[16] Z. Li-Li, H. Lian-Feng and S. Qin-Ying. Research on Requirement for High-quality Model of Extreme Programming. *International Conference on Information Management, Innovation Management and Industrial Engineering*. 2011, 518-522.

[17] K. Stapel, D. Lubke and E. Knauss. Best Practices in eXtreme Programming Course Design. *Proceedings of the 30th International Conference on Software Engineering*, 2008, 771-775.

[18] K. Vlaanderen, S. Brinkkemper, S., Jansen and E. Jaspers. The Agile Requirements Refinery: Applying SCRUM Principles to Software Product Management. *Software Product Management (IWSPM), 2009 Third International Workshop on* 2009, 1-10.

[19] M. J. Akhtar, A. Ahsan and W. Z. Sadiq. Scrum Adoption, Acceptance and Implementation (A Case Study of Barriers in Pakistan's IT Industry and Mandatory Improvements). *Industrial Engineering and Engineering Management (IE&EM), 2010 IEEE 17Th International Conference on* 2010, 458 - 461.

[20] A. I. Khan, R. J., Qurashi and U. A. Khan. "A Comprehensive Study of Commonly Practiced Heavy and Light Weight Software Methodologies." *International Journal of Computer Science and Issues*, 8(4):441-450, 2011.

[21] D. Friis, J. Ostergaard and J. Sutherland. Virtual Reality Meets Scrum: How a Senior Team Moved from Management to Leadership. *System Sciences (HICSS), 2011 44th Hawaii International Conference on* 2011, 1-7.

[22] M. Shalaby and S. El-Kassas. Applying Scrum framework in the IT service support domain. *Services Computing Conference (APSCC), 2011 IEEE Asia-Pacific*, 2011, 9-15.

[23] N. R. Mead, V. Viswanathan and D. Padmanabhan. "Incorporating Security Requirements Engineering into the Dynamic Systems Development Method." *International Journal of Security and Its Applications*, 2(4):67-80, 2008.

[24] T. Dyba and T. Dingsoyr. "Empirical studies of agile software development: A systematic review." *Information and Software Technology*. 1-27, 2008.

[25] P. Tumbas and P. Matkovic. "Agile vs Traditional Methodologies in Developing Information Systems." *Management Information Systems*. 15-24, 2006.

[26] D. Barrit. 2002. "IEC 61131 and DSDM in real-time process control applications." *Computing & Control Engineering Journal*, 13(2), 94-100, 2002.

[27] B. J. J. Voigt, M. Glinz and C. Seybold. Dynamic System Development Method. 1-16, 2004.

[28] V. Szalvay. An Introduction to Agile Software Development. Danube Technologies, Inc, 1-11, 2004.

[29] D. Hartmann and R. Dymond. Appropriate Agile Measurement: Using Metrics and Diagnostics to Deliver Business Value. *Proceedings of the conference on AGILE* 2006.

[30] M. Udo, T. S. Vaquero, J. R. Silva and F. Tonidande. "Lean Software Development Domain." *Association for the Advancement of Artificial Intelligence*.

[31] J. Unger and J. White. Agile user centered design: enter the design studio - a case study. *Proceeding CHI EA '08 CHI '08 Extended Abstracts on Human Factors in Computing Systems*, 2008, 2167-2178.

[32] L. Lindstrom and R. Jeffries. "Extreme Programming and Agile Software Development Methodologies." *CRC Press LLC*, 1-7, 2003.

[33] R. Ramsin and R. F. Paige. "Process-Centered Review of Object Oriented Software Development Methodologies." *ACM Computing Surveys*, 40(1):1-89, 2008.

[34] T. Hayata and J. Han. A Hybrid Model for IT Project with Scrum. *Service Operations, Logistics, and Informatics (SOLI), 2011 IEEE International Conference on* 2011, 285-290.

[35] G. Vanderburg. A Simple Model of Agile Software Processes or Extreme Programming Annealed. *Proceedings of the 20th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications.* 2005, 539-545.

[36] F. Maurer and G. Melnik. Agile Methods: Moving towards the Mainstream of the Software Industry. *Software Engineering, International Conference on* 2006, 1057-1058.

[37] A. C. Nelson and J. T. C. Teng. "Do systems development methodologies and CASE tools decrease stress among systems analysts?" *Behaviour & Information Technology*, 19(4):307-313, 2000.

[38] S. Nerur, R. Mahapatra and G. Mangalaraj. "Challenges of Migrating to Agile Methodologies." *Communications of the ACM*, 48(5):73 -78, 2005.

[39] M. Cristal, D. Wildt and R. Prikladnicki. Usage of SCRUM Practices within a Global Company. *Global Software Engineering, 2008. ICGSE 2008. IEEE International Conference on* 2008, 222 - 226.

[40] R. W. Van Dijk. Determining the Suitability of Agile Methods for a Software Project. *15th Twente Student Conference on IT*. 2011

[41] A. Mahanti. "Challenges in Enterprise Adoption of Agile Methods – A Survey." *Journal of Computing and Information Technology*, 3:197–206, 2006.

[42] D. Turk, R. France and B. Rumpe. Limitations of Agile Software Processes. *In Proceedings of the Third International Conference on Extreme Programming and Flexible Processes in Software Engineering* XP2002.

[43] M. R. J. Qureshi. "Agile software development methodology for medium and large projects." *The Institution of Engineering and Technology*, 6(4):358–363, 2012.

[44] M. Hneif and S. Hock Ow. "Review of Agile Methodologies in Software Development*." International Journal of Research and Reviews in Applied Sciences*, 1(1):1-8, 2009.

[45] Y. B. Leau, W. K. Loo, W. Y. Than and S. F. Tan.). Software Development Life Cycle Agile vs Traditional Approaches. *International Conference on Information and Network Technology*, 2012, 37:162-167.