International Journal of Innovative Research and Advanced Studies (IJIRAS)
Volume 6 Issue 5, May 2019

ISSN: 2394-4404

Comparing Agile And Traditional System Development Methodologies

Dr. Tefo Sekgweleo

Cape Peninsula University of Technology, South Africa

Abstract: ***System development methodologies play a vital role in system development life cycle. They serve as a guideline on how to develop a system. However, it is imperative to know which methodology to adopt and use when carrying out the systems development project. One methodology cannot be a silver bullet to all software development projects within the organization. They are not one size fit all kind of methodologies. Therefore, the literature research methodology was applied for this study to compare both agile and traditional methodologies. This was to gain a better understanding of the agile as well traditional methodologies. It is also imperative to understanding the weaknesses and strengths of these methodologies. As a result the systems development team needs to understand the differences of these methodologies prior to selecting which methodology to follow. These methodologies have various types which consist of different stages.***

Keywords: ***System Development Life Cycle (SDLC), Information System (IS), Software, System, Agile Methodologies, Traditional Methodologies, Software Development Team***

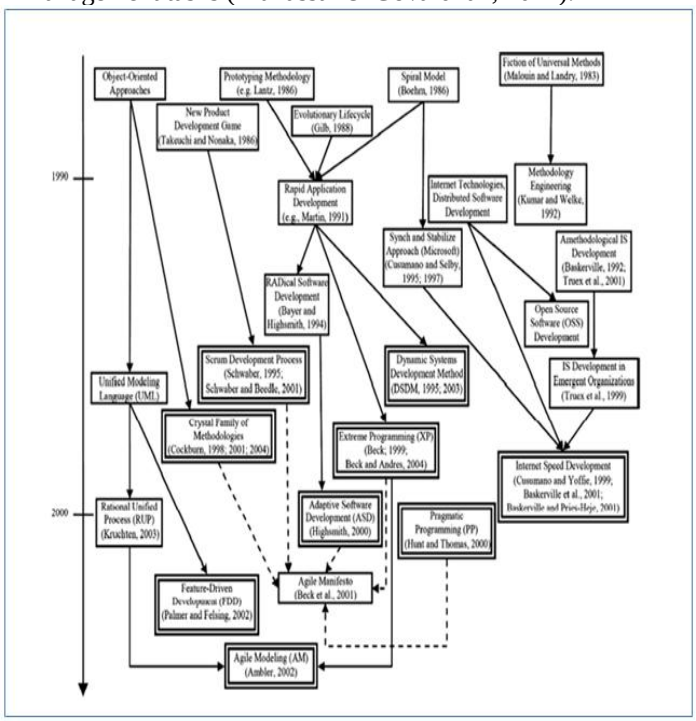
I. INTRODUCTION

In the olden days, software development was focused around a developer to write code in order to solve a problem. Currently, that has changed due to the fact that systems are so huge and complex that teams of architects, project managers, analysts, developers, testers and users have to work together to create the large number of lines of written code that drive the competitiveness of the organisations. Therefore, prior to any software development it is imperative for the software development team to decide which software development method to follow based on the size of the project. Nelson and Teng (2000) define SDLC as a guideline and logical process employed by system developers to develop systems to be used by organisations. Masrek (2008) stipulates that a methodology consists of various stages that help guide systems developers to choose techniques that might be appropriate at each stages of the project and also assist them in planning, managing, controlling and evaluating IS projects.

The systems development methodology also known as SDLC is mainly used in several engineering and industrial fields such as systems engineering, software engineering, mechanical engineering, computer science, computational sciences and applied engineering (Bassil, 2012). According to Fitzgerald (2000) the SDLC was first applied in relation to systems development in this era of Dr Royce around 1970. With his article titled "Managing the development of large software systems", he explained his experiences regarding his involvement in the development of large systems. There are various systems development methods that are designed to be employed during development process of systems which could be either traditional or agile. They all consist of a sequence of steps that must be followed and completed by system developers in order to achieve some results and deliver a final product (Bassil, 2012).

According to Sekgweleo (2015) the traditional systems development methodologies include Waterfall, Spiral, V-Model, Rational Unified Process (RUP) and Rapid Application Development. On the other hand agile systems development methodologies include eXtreme Programming (XP), Scrum, Dynamic Systems Development Method (DSDM) and Lean Software Development (Sekgweleo, 2015). Each method follows a particular life cycle in order to ensure success in process of systems development. These

methodologies also include recommendation or techniques for resource management, planning, scheduling and other management tasks (Munassar & Govardhan, 2011).



Source: Ramsin and Paige (2008)

Figure 1

The development of IS has always been a complex process hence it requires the use of methodological approaches such as SDLC, which is a systematic process for creating a system (Tan & Tan, 2010). Too much planning as well as decision making is needed when developing systems. Sisaye (2005) posits that when it comes to decision situations where information sharing exists, teams are able to produce better decisions than would have been realized through individual decisions, which resulted in improved compensation and reward allocations among the team members. As mentioned earlier, the system development requires a team with specialized skills that will enable it to achieve the objectives of the requested system. The collaboration of such specialized skills is very important.

II. SOFTWARE DEVELOPMENT

Software development is a process of converting the user/business requirements into a system. Satzinger et al., (2004) defines information system as "a collection of interrelated components that collect, process, store and provide as output the information needed to complete a business task". Therefore, to achieve the objective of creating the solution it is vital for the systems development team to have the techniques and tools in place. Avison and Fitzgerald (2006) describes a technique as a way of doing a particular activity in the information systems development process and any particular methodology may recommend techniques (such as flow charts, entity-relations diagram, manual document specifications, organization chart) to carry out many activities. On the other hand, tools include software packages (such as C++, Java, Oracle, Delphi) that helps aids the aspects of information systems (Avison & Fitzgerald, 2006).

System development is the innovation, discovery and artistry and each foray into a development project which presents new and difficult challenges that cannot be overcome with one-size-fits-all, cookie cutter solutions (Szalvay, 2004). Therefore, skills within the systems development team have to be applied to the fullest capacity in order to produce the requested solution. Tumbas and Matković (2006) stipulate that the characteristics of individuals participating in the system development apply the primary influence on the quality of project activities, where the optimal effect is reached by the teamwork. This can be achieved through working together as a unit. Hence, various skills complement each other.

III. SOFTWARE TESTING

Another crucial step that is required after systems development is software testing. It helps verify and validate what the developer has developed. Software testing can be defined as a process of analyzing a software item to detect the differences between existing and required conditions (that is defects/ errors/ bugs) and to evaluate the features of the software item (Waje, 2014). The intention is not only to find defects but to help developers realize the mistakes they have committed and provide recommendations of resolving the mistakes. Software testing also determines whether the software meets the specified requirements and finds any errors present in the code (Munassar & Govardhan, 2010).

This exercise takes place to eradicate any problems during systems development. Systems need to be bug free prior to its implementation. It is very bad for end users to experience hick-ups while using the system. That as well could be very dead-remental to the organization's image. To keep customers and attract new ones it is vital for the organization to develop credible and quality systems. Therefore, the main purpose of testing may be quality assurance, reliability estimation, validation and verification (khan & Khan, 2012).

Software testing is dictated by the systems development methodology selected by the systems development team. The software testing team cannot follow a traditional methodology when agile methodology has been adopted vise vesa. There are certain characteristics to look for when selecting the system development methodology. According to Ahmar, (2010) the methodology selection criteria include the following project time, clarity of user requirements, familiarity with technology, system complexity, system reliability and schedule visibility. Therefore, all the testing activities need to be planned around the selected methodology.

IV. AGILE METHODOLOGY

Basically, agile methodologies are alternatives to traditional methodologies which are usually used in systems development. It helps teams to break large projects into small chunks which could be managed better. It also focuses on keeping the code simple, testing regularly and delivering

functional bits of the system as soon as they are ready. Hence, the goal is to build upon small chunks approved by the clients as the development progresses instead of delivering one large system at the end of the project. According to Sekgweleo (2015) agile methodology can be defined as a "subset of iterative and evolutionary methods that are based on iterative enhancement and opportunistic development processes". Tan and Tan (2010) stipulate that the agile systems development puts a lot of emphasis on early and continuous delivery of the software product.

Therefore, agile methods usually promote a disciplined project management process that encourages regular inspection and adaptation, a leadership philosophy that encourages teamwork, self-organization and accountability, a set of engineering best practices that allow for rapid delivery of high-quality system, and a business approach that aligns development with customer needs and company goals (Chang, 2010). Some of the leading agile development methodologies include Extreme programming (XP), Adaptive Software Development (ASD), Feature-Driven Development (FDD) and Scrum (Meso and Jain, 2006). Nerur, et al., (2005) posits that the mentioned agile methodologies are ideal for projects that display high changeability in tasks due to forever changing requirements in the capabilities of people and in the technology being used.

Mishra and Weistroffer (2008) states that there are six common features to the various agile methods which include collaboration, code reviews, small teams, short release schedules, time-boxing and constant testing. Therefore, the agile team is constantly meeting and working smart to achieve the user requirements. This methodology provides the organization with the ability to rapidly evolve IS solutions (Meso & Jain, 2006). The technology keeps on improving and changing. Therefore, it is vital for organisations to keep up with it by improving its systems in order to remain competitive in the market.

V. TRADITIONAL METHODOLOGY

The traditional systems development methodology has different goals for each stage of development where each stage is completed for the next one to begin. The traditional system development methods are dependent on a set of predetermined processes and on-going documentation which is written as the work progresses and guides further development (Leau, 2012). However, should any changes be deemed necessary on the completed stages, it is possible to log change request and correct that through change management process. The perceived advantages of the traditional process are that it allows for standardisation and managerial control. Deadlines are set for each stage of development. This process leads to the project being delivered on time because each stage has been planned in detail.

According to Matković and Tumbas (2010) the key advantages contributing to the popularity of the model include strictly defined model and it is characterised by standardised activities described in detail in all development stages. These stages consist of six stages including requirement gathering, analysis, design, testing, implementation and maintenance (Jiang, 2009). The mentioned stages have to be well documented and signed off by various stakeholders involved within the development of the system.

According to Bassil (2012) some of the successful SDLC models include the Waterfall, spiral, incremental, rational unified process (RUP) and rapid application development (RAD). However, the mentioned methodologies consist of various stages. Each stage is separate and team members involved in a stage needs to ensure that the stage is perfected prior to delivering to next stage in order to deliver a greater project output (Amlani, 2012). The limitation with this methodology is that the user has to wait too long get to experience the system and use it. The only thing they are exposed to is just too much documentation. However, once testing is complete the users will have to go through user acceptance testing to verify whether what is delivered is what was agreed upon. Therefore, it starts with an idea or opportunity that needs to be evaluated to determine how feasible it is. If it proves to be feasible the team is put together and it has to decide on which methodology to follow prior to software development.

VI. AGILE VS TRADITIONAL

Agile Methods rely on informal communication rather than huge documentation to rapidly spread information throughout the team and to other stakeholders (Mishra & Weistroffer, 2008). On the other hand traditional software development methodologies require defining and documenting a stable set of requirements at the beginning of a project (Leau, 2012).

The traditional methodologies may be disappointing when it comes to projects whereby system requirements change regularly and the development schedules have to be shortened (Sekgweleo, 2015). However, agile development is very flexible and tends to create a room for frequent requirement changes to the system (Tumbas & Matković, 2006).

Agile also provides limited support for the development comprising of large teams because it forces team members to be located in different locations (Sekgweleo, 2015). The traditional methodology is very effective for team members located in different locations (Amlani, 2012). This methodology becomes handy for organizations that outsource the systems development to off-shore countries.

In the traditional methodology users keep on adding requirements which increase the software development time and expense (Amlani, 2012). Users may not be aware of all the requirements at the beginning of the project. Amlani (2012) stipulates that in agile the joint application design (JAD) session help reduce development time which reduces the labour cost for both developers and users, it reduce the involvement time of business experts which reduces the cost further and also the cost reduced by catching errors, misunderstandings and mistakes early in the development stage.

	AGILE	TRADITIONAL
User requirement	Iterative acquisition	Detailed user requirements are well-defined before coding/implementation
Rework cost	low	high
Development direction	Readily changeable	Fixed
Testing	On every iteration	After coding phase completed
Customer involvement	high	low
Extra quality required for developers	Interpersonal skills & basic business knowledge	Nothing in particular
Suitable Project scale	low to medium-scaled	Large-scaled

Source: Leau et al., (2012)

Table 1

Amlani (2012) posits that in traditional methodology team members of the other to be involved in other stages sits idle except they are under the current working stage. Within the agile project, it is vital to have frequent interactions between users, development teams and all the other interested parties in development (Tumbas & Matković, 2006). Therefore, all the project stakeholders are involved daily.

With traditional methodology the amount of documentation is excessive and inflexible and each stage that is completed needs formal review and detailed documentation development (Munassar & Govardhan, 2010). Requirements gathering take too long to be documented due to consulting a number of users involved in the project. On the other hand with the agile methodology requirements are documented as stories on the story cards and have to be included in a release are determined by the time available and their relative priority (Munassar & Govardhan, 2010). It is quick to document requirements because they (requirements) are limited to a particular iteration.

It is the responsibility of the software development team to choose a suitable software development methodology for a particular project. Software development methodologies are not one size fit all. They need to be applied properly. Both agile and traditional methodologies, when applied wrongly they will ultimately fail the software development team.

VII. RESEARCH METHODOLOGY

There are various research methods that could be applied when conducting a research. Literature research methodology is one of those methods. Its substantial variance from other methodologies is that, it helps the researcher to indirectly access information from a variety of literatures, which is generally referred to as "non-contact method" (Lin, 2009). Therefore, literature research methodology was used in this study to read through, analyze and categorise literatures in order to identify the essential attribute of materials (Dotong & Laguador, 2015).

The researcher reviewed the existing literature around agile and traditional methodologies in order to have a clear understanding between these methodologies. A number of peer reviewed journals including chapter in books were reviewed. The researcher discovered that one methodology cannot be a silver bullet for all software development projects. Both methodologies have got their own weaknesses and strengths. Therefore, a methodology can be chosen based on the nature of the software development project and not on which methodology is better than which.

Lately, organisations are using the term agile as if it is new methodology or a solution to all the problems that are encountered in the software development projects. Both agile and traditional methodologies have been around for quite some time. Therefore, any method (irrespective of whether is agile or traditional) may be used based on how suitable it is for the software development project to be carried out. Due to such hype, the researcher saw it fit to conduct a study to compare both methodologies.

VIII. FINDINGS AND DISCUSSION

Both the agile and traditional methodologies have their own strengths and weaknesses. After having reviewed those strengths and weakness the researcher realized that the strengths of agile methodologies are the weaknesses of traditional methodologies and vice versa. Therefore, it starts with an idea or opportunity that needs to be evaluated to determine how feasible it is. If it proves to be feasible the team is put together and it has to decide on which methodology to follow prior to software development.

In this regard the characteristics of individuals participating in the system development apply the primary influence on the quality of project activities, where the optimal effect is reached by the teamwork. This can be achieved through working together as a unit. Hence, various skills complement each other.

Therefore the methodology selection criteria will have to include the following: project time, clarity of user requirements, familiarity with technology, system complexity, system reliability and schedule visibility. Hence, all the testing activities will prerequisite to be planned around the selected methodology.

Therefore, it is imperative for the software development team to know and understand various (agile and traditional) methodologies. They need to know and understand which one to adopt and when. Therefore, it is incorrect to blame the software development methodology when things are not favorable to for the software development project. A methodology is just a framework or guideline that enables the software development team to achieve the goals of the software development project.

IX. CONCLUSION

Software methodologies are not one size fit all. Software development projects are different and it would not be wise to apply the same methodology throughout different projects. Lately there is a hype within organizations about going agile as if agile is the solution to all its software development projects. It is imperative to take into cognizance that various methodologies are suitable for different software development projects. Having to follow the same methodology to different software development projects, the organization may fall into a trap whereby some of its projects fail due to the limitations of the methodologies.

Both agile and traditional methodologies have strengths and limitations. Therefore, it will be incorrect to say one

methodology is better than the other. Looking at the comparison, one is able to spot that what is a limitation on the traditional method it turns into strength on the agile method. As a result, the limitation on one method gets improved on the other methodology. The choice of following a particular methodology is determined by the size and the nature of the project. Methodologies are not one size fit all. Hence, organizations are trapped in basing (their) its strategies, standards and procedures upon a particular methodology. This forces them to follow a single methodology for all the projects they carry out.

It is just like automobile industry, the mistakes identified on the predecessor car get to be fixed on the successor car. Therefore, lessons learned from the previous system may help the organization to develop a better system in future. Provided such organization conduct audits after each and every systems development and note those down in order (to improve on them in future) do things better in future. In a nut shell one could argue that weaknesses identified in the traditional methodologies were improved during the introduction of agile. The strengths of traditional methodologies became agile weaknesses. Therefore, both methodologies struck a balance amongst each other.

REFERENCES

[1] Bassil, Y. 2012. A Simulation Model for the Waterfall Software Development Life Cycle. International Journal of Engineering & Technology (iJET), 2(5): 2049-3444.
[2] Dotong, C.I. & Laguador, J.M., 2015. Philippine Quality Assurance Mechanisms in Higher Education towards Internationalization. Studies in Social Sciences and Humanities, 3(3): 156-167.
[3] Fitzgerald, B. 2000. Systems development methodologies: the problem of tenses. Information Technology & People, 13(3):174-185.
[4] Lin, G. 2009. Higher Education Research Methodology-Literature Method. International Education Studies.
[5] Leau, Y. B, Loo, W. K., Tham, W. Y. & Tan, S. F. 2012. Software Development Life Cycle AGILE vs Traditional Approaches. International Conference on Information and Network Technology.
[6] Meso, P. & Jain, R. 2006. Agile Software Development: Adaptive Systems Principles and Best Practices. Information Systems Management, 19-30.
[7] Munassar, N. B. A. & Govardhan, A. 2011. Comparison between Traditional Approach and Object-Oriented Approach in Software Engineering Development. IJACSA) International Journal of Advanced Computer Science and Applications, 2(6):70-76.
[8] Munassar, N. B. A. & Govardhan, A. 2010. A Comparison Between Five Models Of Software Engineering. IJCSI International Journal of Computer Science Issues, 7(5):94-101.
[9] Matković, P. & Tumbas, P. 2010. A Comparative Overview of the Evolution of Software Development Models. International Journal of Industrial Engineering and Management (IJIEM), 1(4):163 - 172.
[10] Mishra, S. & Weistroffer, H. R. (2008). Issues with Incorporating Regulatory Compliance into Agile Development.
[11] RAMSIN, R & PAIGE, R. F. 2008. Process-Centered Review of Object Oriented Software Development Methodologies. ACM Computing Surveys, 40(1).
[12] Sekgweleo, T. 2015. Understanding Traditional Systems Development Methodologies. International Journal of Advances in Management and Economics, 4(3):51-58.
[13] Sekgweleo, T. 2015. Understanding Agile System Development Methodologies. International Journal of Advanced Research in Computer Science and Software Engineering, 5(7):18-24.
[14] Sisaye, S. 2005. Teams and management control systems: a synthesis of three organizational development approaches. Leadership & Organization Development Journal, 26(3):172-185.
[15] Rob, M. A. 2006. Dilemma Between the Structured and Object-Oriented Approaches to Systems Analysis and Design. Journal of Computer Information Systems, 32-42.
[16] Tan, W. K. & Tan, C. H. 2010. Teaching Information Systems Development via Process Variants. Journal of Information Systems Education, 21(2):159-172.
[17] Nerur, S., Mahapatra, R. & Mangalaraj, G. 2005. Challenges of Migrating to Agile Methodologies. COMMUNICATIONS OF THE ACM, 48(5):73-78.
[18] Germain, E. & Robillard, P. N. 2005. Engineering-based processes and agile methodologies for software development: a comparative case study. The Journal of Systems and Software 75:17–27.
[19] Waje, S., Gaikwad, V. & Chaudhari, P. 2014. Software Testing, Mythology & Methodologies. International Journal of Emerging Technology and Advanced Engineering, 4(2):673-677.
[20] Khan, M. E. & Khan, F. 2012. A Comparative Study of White Box, Black Box and Grey Box Testing Techniques. IJACSA) International Journal of Advanced Computer Science and Applications, 3(6):12-15
[21] Szalvay, V. 2004. An Introduction to Agile Software Development.
[22] Chang, M. 2010. An Agile approach to library IT innovations. Library Hi Tech, 28(4): 672-689.
[23] Masrek, M. N., Hussin, N. & Tarmuchi, N. 2008. An exploratory study on systems development methodologies for web-based applications. Information Management & Computer Security, 16(2):137-149.
[24] Jiang M, Jong CJ, Poppell P, Budhathoky, K, Hull R (2009) System Infrastructure Development Life Cycle for Enterprise Computing Systems.
[25] Amlani, R. D. 2012. Advantages and Limitations of Different SDLC Models.International Journal of Computer Applications & Information Technology, 1(3):6-11.
[26] AVISON, D. & FITZGERALD, G. 2006. Information Systems Development Methodologies, Techniques & Tools. 4th ed. United Kingdom: McGraw-Hill.
[27] Satzinger, J. W., Jackson, R. B. & Burd, S. D. 2004. Systems Analysis and Design in a Changing World. 3rd ed. USA: Thomson.

[28] Tumbas, P. & Matković, P. 2006. Agile vs Traditional Methodologies in Developing Information Systems. Management Information Systems, 15-24.
[29] Leau, Y. B., Loo, W. K., Tham, W. Y. & Tan, S. F. 2012. Software Development Life Cycle AGILE vs Traditional Approaches. 2012 International Conference on Information and Network Technology (ICINT 2012).
[30] Ahmar, M.A.A. 2010. Rule based expert system for selecting software development methodology. Journal of Theoretical and Applied Information Technology, 143-148.
[31] Munassar, A. M. A. & Govardhan, A. 2010. A Comparison Between Five Models Of Software Engineering. IJCSI International Journal of Computer Science Issues, 7(5):94-101.
[32] http://mos.sciedupress.com