Table of Contents

ABSTRACT

The exponential rise in the demand for mobile application has triggered a huge rise in the number of mobile application development projects. However, adaptive software development method is required to fulfill the highly volatile requirements of mobile applications. Many analysts and researchers have identified the Agile approach as a natural fit for the development of mobile applications. Thus, it's very important to explore the numerous Agile methodologies that can be employed in the development of mobile applications. This paper extensively analyze how the adoption of Agile approaches, can help to improve the development of software in general, and mobile applications in particular. The findings show that many Agile methods, have the capacity to deliver enhanced speed and quality for mobile application development

INTRODUCTION

Software development is a generic term, used in reference to all the computer programming, testing, documenting and bug fixing involved in the creation and maintenance of computer applications. Generally, software development does not only involve the writing of codes, but include; the conception of the desired software through to the final manifestation of the software. This is usually executed in a planned and structured process. Thus, software development may include prototyping, research, new development, reuse, re-engineering, maintenance etc.

On today's digital world, software development and application, is progressively on the increase. The use of software has been successfully merged with many numerous fields of human endeavors, a situation that has help to boost its complexity. The process of developing software is also being made more difficult by the dynamic nature of customers' demand. The old methodology employed in software development cannot be used to satisfy the latest requirements of the market in the most suitable way.

The aforementioned challenges have lead to the generation and subsequent adoption of new methods of software development, which are capable of solving this problem. One such example is the agile methodology. Like every other new methodologies, the agile methodology include specific modifications, specially designed to make the whole process of software development, to become more flexible and productive. In this paper, we comprehensively discuss the numerous ways, through which agile has effectively changed the process of software application development.

HISTORY OF SOFTWARE DEVELOPMENT APPROACHES

Computer programming is the building block of software development. It was in 1954 that computer programming first started, with the emergence of such structured languages like Fortran (Maurer and Martel, 2002). By the 1960s, the programming process has evolved to object oriented languages in the (Maurer and Martel, 2002). During the same period, the methodologies employed in software development also changed progressively.

Normally, it's the chosen methodology to be adopted in software development that guides the developer throughout the process. Thus, the software developer must first of all identify the exact methodology to use for the process. Generally, the methodology to be used in software development consists of several phases, which must be followed sequentially in course of the development process. Each phase has specific guidelines, which must be applied in the process.

The first high quality methodology for software development, came into existence after the software crisis in the 1970s (Klimeš and Procházka, 2006). The structured methodologies for software development were employed by software engineers to minimize the destructive effects of the software crisis. Normally, structured methodologies split the process of software development into phases. When using such methodologies, the software developer has to concentrate on one phase at a time. The practice of following the phase sequentially helps in minimizing the number of failed or uncompleted software projects. This also helps to cut down the overall cost as well as the developmental time of software projects. The net result will be an overall reduction of the effects of software crisis is limited (Klimeš and Procházka, 2006).

The structured methodologies for software development were succeeded by Object-Oriented methodologies. Common examples of these methodologies include: Unified Process and Rational Unified Process (RUP). They include not only the objects and object oriented principles, but also the best practices from structured methodologies (Klimeš and Procházka, 2006).

AGILE METHODS

Overview

The Agile methodologies are software development tools, which are specifically based on opportunistic development processes and iterative enhancement (Salo and Abrahamsson, 2008). This unique methods are functionally and structurally iterative and evolutionary (Cho, 2008). Generally, there are four features, which are fundamental to the practical applications of all agile methodologies. These include: iterative and evolutionary development, adaptive planning, rapid and flexible response to change and promotion of communication (Maher, 2009). The main focus is on ensuring the methodologies are in line with the principles of "Light but sufficient", while still being communication-centered and people-oriented.

Generally, Agile methodologies are more suitable for executing small projects (Zuo et, 2010). This explains why many analysts described it as lightweight process. In Agile software development, the developer normally start the process by tackling simple and predictable approximations, before proceeding to the more detailed part of the project. This progression usually involved increasing the details of some requirements throughout the life of the development. This incremental requirements refinement further refines the design, coding and testing at all stages of production activity (Begel, A. and Nagappan, 2007). This is in line with the stated principle of Agile software development, which proposes that; "at regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly" (Mordinyi et al, 2010). Through this way, the whole process will eventually lead to the production of software that will be very accurate and useful.

Thus, Agile methodology is specifically used to overcome the challenges posed by an unpredictable, disordered business and technology environment (Mordinyi et al, 2010). This functionality enables developers, making use of this methodology to achieve higher quality software within a short period of time (Ahmed et al, 2010). It also helps in the self organization of teams, customer collaboration, less documentation and reduced time to market (Ahmed et al, 2010).

The Agile methods of software development consist of several lightweight methodologies. Wang et al (2009) listed the common examples of these as Scrum, Crystal Clear, Extreme Programming (XP), Adaptive Software Development (ASD), Feature Driven Development (FDD), and Dynamic Systems Development Method (DSDM) Crystal, Lean Software Development etc. Each of these methods has its own unique features and prospective applications. Nevertheless, all the aforementioned Agile methodologies, basically work on the same concept.

Earlier on, we listed iterative and evolutionary development as important features of the Agile methodologies. Generally, the Agile methodologies involved the splitting of a huge task into smaller increments with planning. These smaller increments are what we know as Iterations. The time duration for iterations is relatively short and can just last for one to four weeks. Each of the iteration process consists of a team, whose members are fully involved in the full software development cycle. The normal software development cycle include: planning, requirements analysis, design, coding, unit testing, and acceptance testing. This sequential process helps to reduce the overall risk associated with the project development. It also enables the developer to implement any new idea that may be conceptualized in course of the project development. Most of the agile implementations use a formal daily face-to-face communication among team members. During such brief communication, the members of the team can share their daily accomplishments, as well as the goals they hope to achieve in the next phase. This communication process enables team members to come up with the most suitable way of tackling the challenges militating against their success. This is because; whenever a problem is shared among the members of a particular team learns something new about is.

Background

The background of Agile methodologies can be traced to the Agile Manifesto, which was constituted in 2001. It’s these manifesto that defined the approach of agile software development. This approach is summarized into twelve basic principles, which are as follows (Devedzic and Milenkovic, 2011; Sohaib, and Khan, 2010):

- Prioritizing customers’ ultimate satisfaction, through quick and continuous delivery of precious software.
- Ability to accept and implement any new idea that is conceptualized in course of the software development, irrespective of the developmental stage, at which the new idea is conceptualized. Generally, the methodological process employed in Agile software development makes it possible to harness change for customer’s competitive advantage.
- Ability to regularly develop and introduce working software. The duration behind successive deliverance can last from just few weeks to few months. However, the shorter the timescale, the more preferable the process.
- Ability for the software developers to work directly with business people, throughout the developmental stage of the software.
- Ensuring the conceptualization and subsequent development of projects around motivated individuals. This is normally accomplished by providing conducive environment as well as support, the individuals need in course of the development. It also involves trusting them to get the job done.
- The face-to-face conversation is the best way to convey information among team members involved in a particular developmental project. It’s efficient, effective and reliable. Thus, it must be integrated into the agile methodologies.
- The fundamental way to assess the progress that has already been attained is to obtain working software.
- Sustainable development can easily be achieved with the Agile methodologies. This is because; the Agile process enables the developers, users and sponsors to maintain a constant pace indefinitely.
- Agility can be greatly enhanced by paying close attention to good design and technical excellence.
- Simplicity is very essential. In this case, it can be referred to as the art of maximizing the quantity of work that is not done.
- It’s the self-organizing teams that can deliver the best architectures, designs and requirements.

- The team regularly identifies how to become more effective and subsequently tunes and adjusts its operational process accordingly.

POPULAR AGILE METHODOLOGIES

We have previously listed the major types of Agile methodologies which include; Extreme Programming (XP), Scrum, Feature Driven Development (FDD), Crystal Method among others. This section briefly discusses four top Agile methodologies, commonly use in Software development.

Extreme Programming (XP)

According to Kumar and Bhatia (2012), the main objective of Extreme programming is to boost the software quality as well as respond adequately to the customers' requirements, which is constantly on the change. There are five major ways, through which the use of Extreme Programming typically improve a software project. These include: Planning, Managing, Coding, Designing and Testing.

Planning involves the splitting of the project into sub-stages, commonly known as iterations. Each iteration process commences with its own planning and the User Stories subsequently written. The main reason for this planning section is to create release schedule, which will help in making frequent small releases.

Managing basically involves the dedication of an open work space to specific teams. Each day, a stand up meeting will be held. During such meetings, the team members will evaluate the project velocity and assess the progress made so far. Any error that is identified will be quickly fixed.

Coding is all about writing of codes. This is actually the building block of the Software development. Apart from writing the code, the test driven development code can also be used in this phase. According to Kumar and Bhatia (2012) "All production code is pair programmed; only one pair integrates code at a time, continuous integration is made".

The designing stage involves the selection of a system metaphor. The inclusion of functionality is usually avoided at the early stage of the process. The design has to be refactor, whenever and wherever possible.

Testing is the final beneficial feature. Here, all the codes involved in the software development are unit tested. Of course, the code must pass the test before being released into the market. If any bug is discovered, tests are created and acceptance tests subsequently run. The final score of the whole process will then be published.

Scrum

Scrum is another agile methodology that is popularly used in software development. It can also be used in the development of other products. This unique agile methodology is most suitable for projects that have aggressive deadline with complex requirements. It's best for projects with an appreciable degree of uniqueness.

Structurally, the scrum process is a general-purpose project management framework. Under this methodology, the overall process is spit into series of iterations popularly known as sprints. The duration for each sprint can last between 2-4 weeks. A scrum team typically consists of just five to nine members, even though the project executable under this method can scale into the hundreds. Such traditional software engineering roles like programmer, tester, designer or architect are not included in the team. The product is owned by the project's main stakeholder. The ScrumMaster is responsible for making sure the team is as productive as possible.

Feature Driven Development (FDD)

Livermore (2007) defined the Feature Driven Development as pragmatic software process, which is both client and architecture centric. There are five main activities, which can be performed iteratively with this great tool. First and foremost, the Feature Driven Development enables the development of an overall model, whose initial result could be classified as a high-level object model and notes. During the usage of this methodology, the preliminary goal is the identification and subsequent understanding of the fundamentals of the exact domain, which the system is trying to address. It's very much possible and acceptable to refine or update this domain at any point in time, throughout the duration of the project. Such a change will always reflect on the Software that is being developed.

The second activity is the development of a feature list. Here, the features of the proposed software are grouped into related sets. The third activity is simply known as plan by feature. This involves the identification of feature set owners as well as class owners. Design by feature is the fourth activity and it’s all about detailed modeling. The final activity is the build by feature, where the system is programmed tested and packaged.

Crystal Method

The Crystal Methods is a wonderful agile software development methodology, which was developed by a great IT professional known as Alistair Cockburn. In this case, the developers concentrate more on the software development at the expense of processes and tools (Ahmed, 2010). Structurally, the method consists of toolkit of methodology elements specifically meant for individual projects. Large or safety critical projects require more methodology elements than small non-critical projects. Crystal Methods give organizations great degree of flexibility, which enables them to use and develop as much methodology as their business needs demand. It is most suitable for small projects that aren’t life critical.

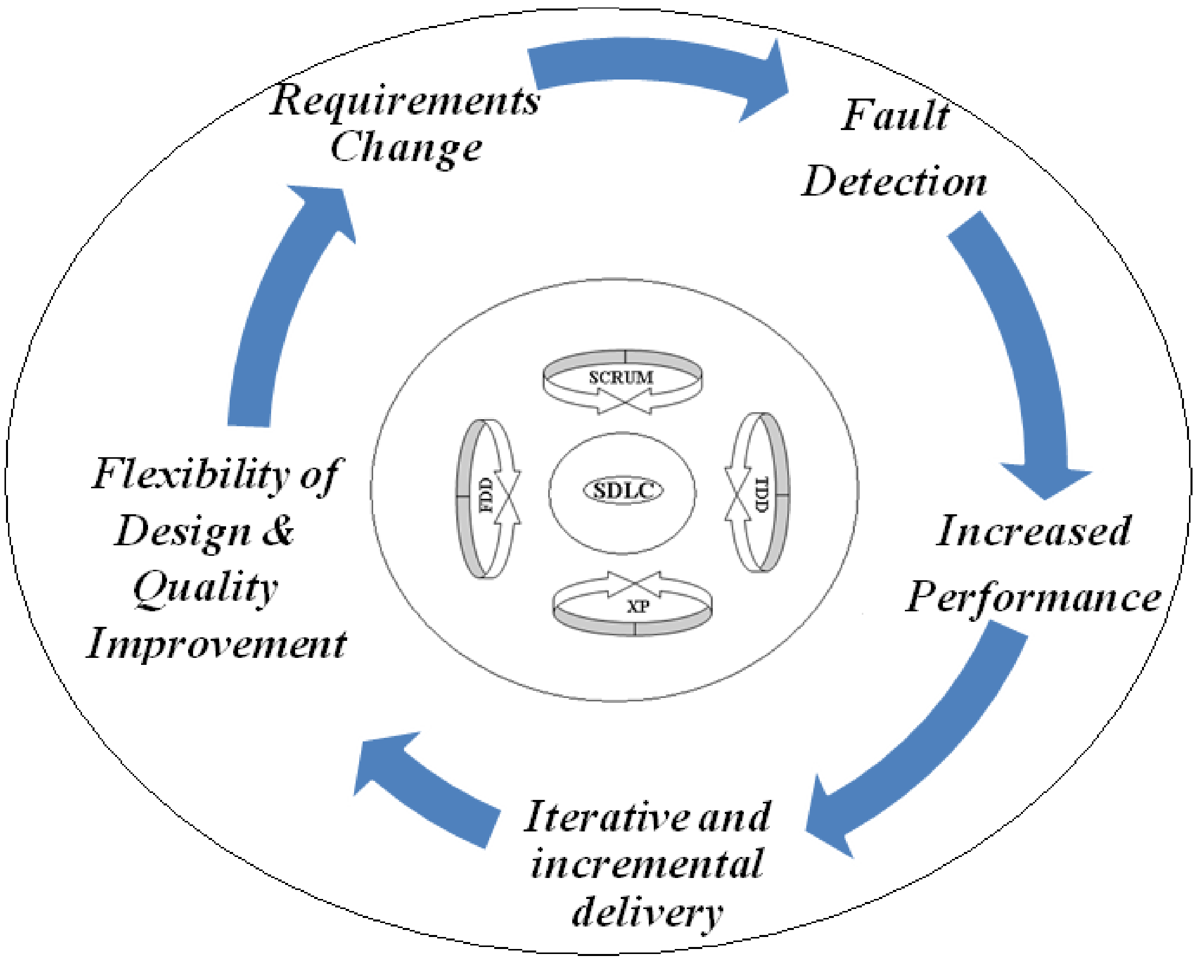
HOW AGILE HAS CHANGED THE SOFTWARE DEVELOPMENT PROCESS

As already stated earlier on, the old methodology employed in software development cannot be suitably used to satisfy the latest requirements of the market. It's these challenges that prompted the emergence of methodology for software development, among which is Agile. Today, the extensive adoption of Agile has greatly changed the software development process. The major ways, through which this methodology has affected this sector of Information Technology, are discussed below.

Handling Change of Requirements

The adoption of Agile methodology has greatly improve the planning phase of Software development. This is because; its implementation has enable customers to become directly involved in the development process. Thus, customers now have capability to improve the control the developmental process of the project. This is normally accomplished through on-site interaction. Such interaction helps the developers to identify the contemporary needs of the end users and update their plans accordingly (Maher, 2009).

Figure 1: Agile Software Development Methodologies with Benefits


Requirements Change
Fault Detection
Increased Performance
Iterative and incremental delivery
Flexibility of Design & Quality Improvement
SCRUM
FDD
SDLC
TDD
XP

Source: Kumar and Bhatia, 2012

Fault Detection

The Agile developmental strategy involves the splitting of the whole process into stages, popularly known as iteration. Normally, testing is always executed at the end of each stage. This tactic enables developers to detect errors at a very early stage and immediately fix them, before it increases in severity. It's practically impossible to accomplish this feat with a plan-driven process model. Similarly, regular testing enables continuous testing feedback, which can be used to improve the functionality of the codes, to be used in future iterations (Wang, 2011).

Increased Performance

Agile methodology has also lead to increased performance of the final product. This can be attributed to the daily standup meetings, which is a characteristic feature of every Agile methodology. These meetings allow all stakeholders, involved in the software development, to exchange valuable information and continuously fine tune its improvements. This feature also promotes teamwork among all members of the team. This is because, good communication among team members facilitates knowledge sharing, team morale and self organizing teams. It also promotes trust among the team members. All these boost the team productivity and generate better performance in terms of good Return on Investment than the sum of all individual output (Nakki et al, 2011).

Iterative and incremental delivery

Using Agile methodology enables the splitting of the project delivery is into small functional releases or increments. This enables stakeholders to effectively control the risks associated with their project. It also helps the stakeholders, to obtain early feedbacks from the customers. These small releases are delivered on a schedule using iterations that typically last between one and four weeks each. During software development, the frameworks are first developed and then updated incrementally to suit the prevailing needs of the organization. Here, the frameworks we are talking about include: plans, requirements, design, code and tests. This strategy enables developer to check and monitor the software functionality, on regular basis (Nakki et al, 2011).

Flexibility of Design

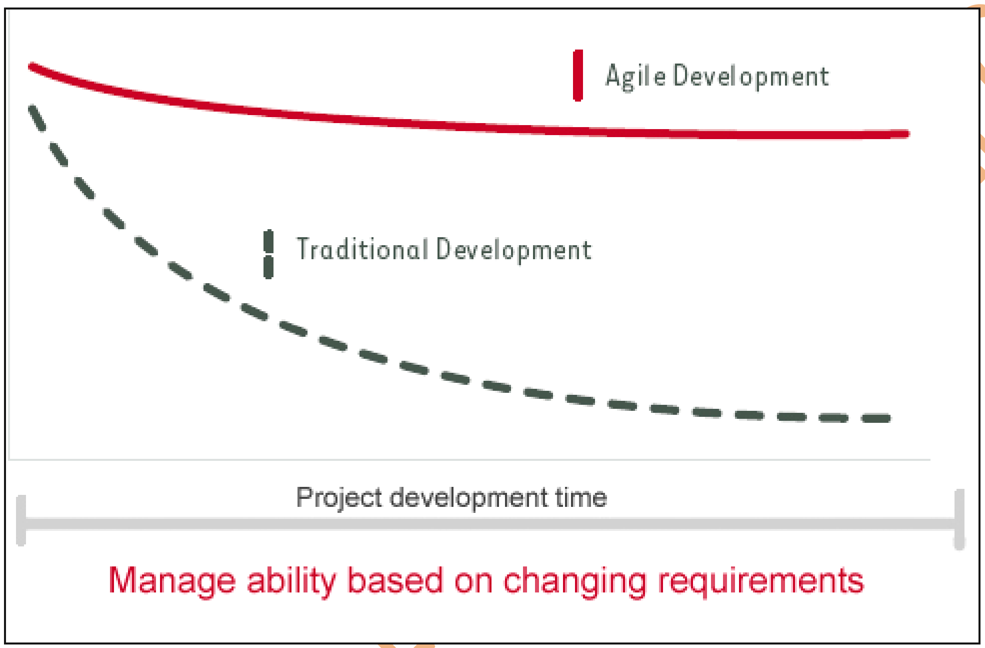
Flexibility simple means the ability to instantly change directions, to fulfill the prevailing needs of the market. The use of Agile methodology has made the process of software development to become more flexible. This shouldn't be a surprise as flexibility is one of the inherent features of Agile methodology. Thus, its adoption in software development has enable developers to handle changes easily. Flexibility is based on the development process used for the project (Nakki et al, 2011; Wang, 2011).

Improvement in Quality

The use of Agile methodology has also lead to an unprecedented increase in the quality of the products that are eventually obtained. This is hugely attributed to the use of refactoring and test-driven development. Refactoring also leads to higher incidents of code re-usage, which usually result to production of better quality of software. In a nutshell, the use of Agile method increase the quality of all aspects of software, ranging from design its operational performance (Nakki et al, 2011; Wang, 2011).

A COMPARATIVE STUDY ON AGILE AND NON-AGILE SOFTWARE DEVELOPMENT PROCESS

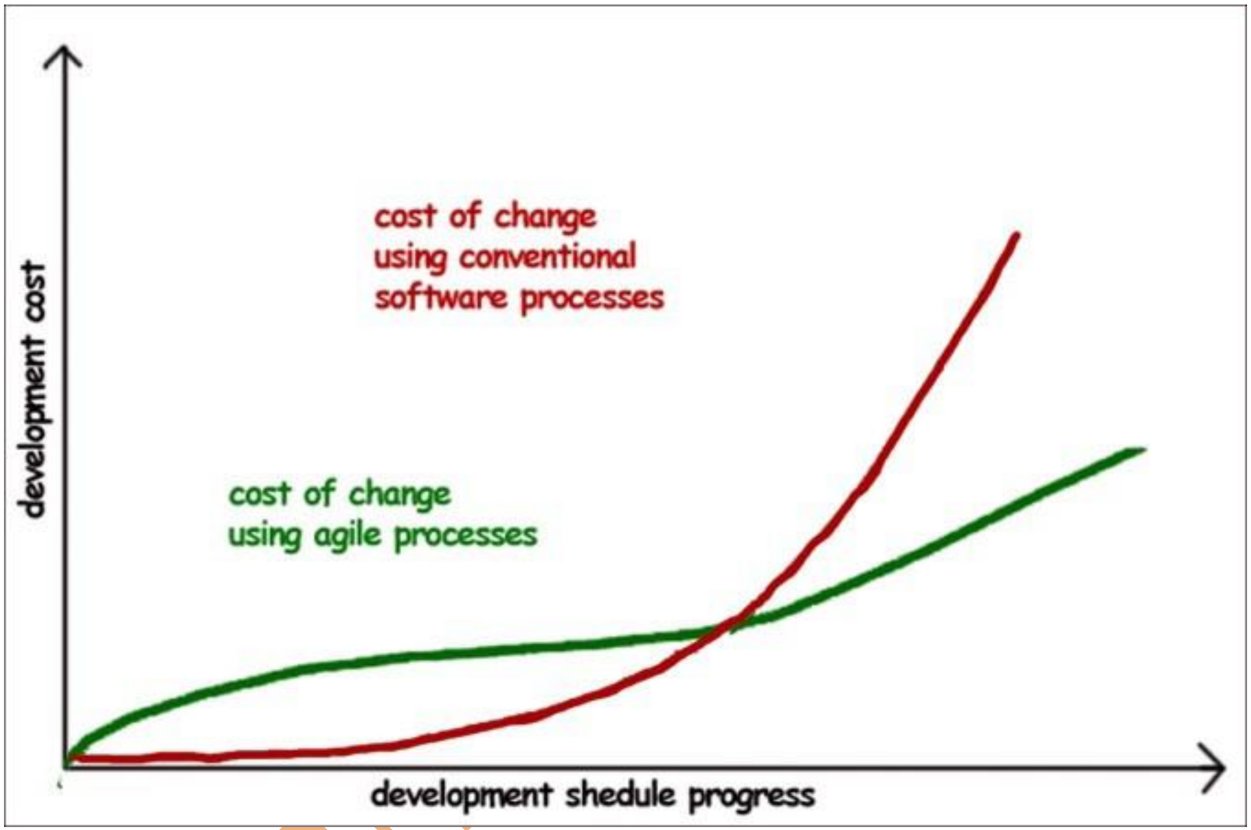
Figure 2: Agile vs. traditional requirements change management



Source: www.versionone.com

The Agile Software Development Method greatly differs from the Non-Agile Software Development Process. According to Dingsøyr et al (2012), the main objective of Agile Software Development Method is on quick delivery of high quality products, while that of Non-Agile Software Development focus mainly on high assurance. An analytical study conducted by Nerur, et al (2005), revealed seven issues to differentiate traditional and agile development. According to the authors, the use of Agile Software Development lead to the development of high quality adaptive software, through the principles of continuous design improvement and testing based on rapid feedback and change, while Non-Agile Software Development method lead to the development of systems that are fully specifiable, predictable, and are built through meticulous and extensive planning. In their own study, Dyba and Dingsoyr (2009) based the analysis of the difference between Agile Software Development Method and Non-Agile Software Development on the unpredictability of the world, as well as emphasizing the value competent people and their relationships bring to software development. The results showed that the Agile Software Development Methods focus mainly on addressing the challenges associated with the unpredictability of the world.

Figure 3: Cost of change for agile and conventional development process



Source: Moniruzzaman and Hossain, 2010

Lots of researchers have successfully conducted some studies, specifically aimed at understanding the differences between Agile Software Development Method and Non-Agile Software Development Methods (Nerur, et al, 2005; Leffingwell, 2007; Lemétayer, 2010; West et al, 2010). The major findings of these authors are summarized.

- **Development Life Cycle**: The development life cycle employed in Non-Agile Software Development is normally linear, while the Agile approach make use of Iterative and evolutionary-delivery model.
- **Style of development**: The style of development used in Non-Agile approach is naturally anticipatory, while that used in Agile-approach is Adaptive.

- **Management :** Under the Non-Agile approach, the managerial procedure is process centric; Command and Control; while the Agile approach is people- Centric Leadership and Corporation.
- **Change:** Whereas the Non-Agile approach tends to be change averse, the Agile methodology readily embraces change
- **Team Organization:** In Non-Agile Software Development Methods, the team organization is generally pre-structured. On the other hand, the team organization Agile Software Development Methods Self Organizing team
- **Client Involvement:** Under Non-Agile Software Development approach, the level of clients' participation is relatively low. This is in contrast to the Agile Software approach, where clients' are considered as a team member.
- **Software Development Process**: The Non-Agile methodology adopts a universal approach and solution in the provision of predictability and high assurance, while the Agile methodology make use of a flexible approach adapted with collective understanding of contextual needs. This helps developer to achieve faster developmental process.

MOBILE SOFTWARE DEVELOPMENT

Over the last decade, the mobile industry has recorded an unprecedented increase, with a complementary rise in mobile applications. The increasing demand for mobile application has driven up the number of projects for mobile application development services. Mobile software development is all about the creation of special software, which can be used in such small, low-power handheld devices like mobile phones. This software includes the factory pre-installed applications on mobile phones and the numerous applications that can be downloaded from app stores and other mobile software distribution platforms. The highly dynamic nature of customers' need as well as technical constraints has continued to militate against the success of mobile apps development. These challenges have prompted mobile apps developers to adopt agile methodology in the quest to provide high quality service delivery. Indeed, the adoption of Agile Methodology has opened a relatively new approach to software development.

How agile can improve Mobile Software Application

Studies have shown that the Agile approaches is highly suitable for the development of Mobile applications (Kim, et al, 2009; Khambati et al, 2008; Flora et al, 2014). This is because of the numerous values and advantages that are realizable from the implementation of Agile practices in mobile application development. According to Kim, et al (2009), the Agile development methods is a natural fit for mobile apps. Flora et al (2014) listed the most significant ways, through which Agile practices improve the development of mobile app projects as follows:

- Agile is hugely compatible with the high volatile requirements of mobile apps.
- The methodology also encourages all stakeholders to partake in the development of the mobile apps.
- The adoption of Agile methodology boosts reliability, a feature that leads to continued use of mobile apps
- Agile development empowers user experience for mobile apps

- This methodology is the most suitable solutions to the incomplete requirement nature of most mobile projects.
- Agile methodology software development is also the best solutions for the experimentation and adaptation of mobile apps.
- It helps developers to identify risks and errors at the early stage of the software development
- Agile is best suitable for quick delivery and short development lifecycle of mobile apps.

AGILE DEVELOPMENT FOR MOBILE APPLICATIONS

As we have already seen, there are numerous values and advantages that are realizable from the implementation of Agile practices in mobile application development. However, this method is a relatively new approach to software development, becoming wide-spread in the last decade. The real life applications of Agile methods can be seen in the Mobile D-project and Hybrid Method Engineering.

Mobile D Process

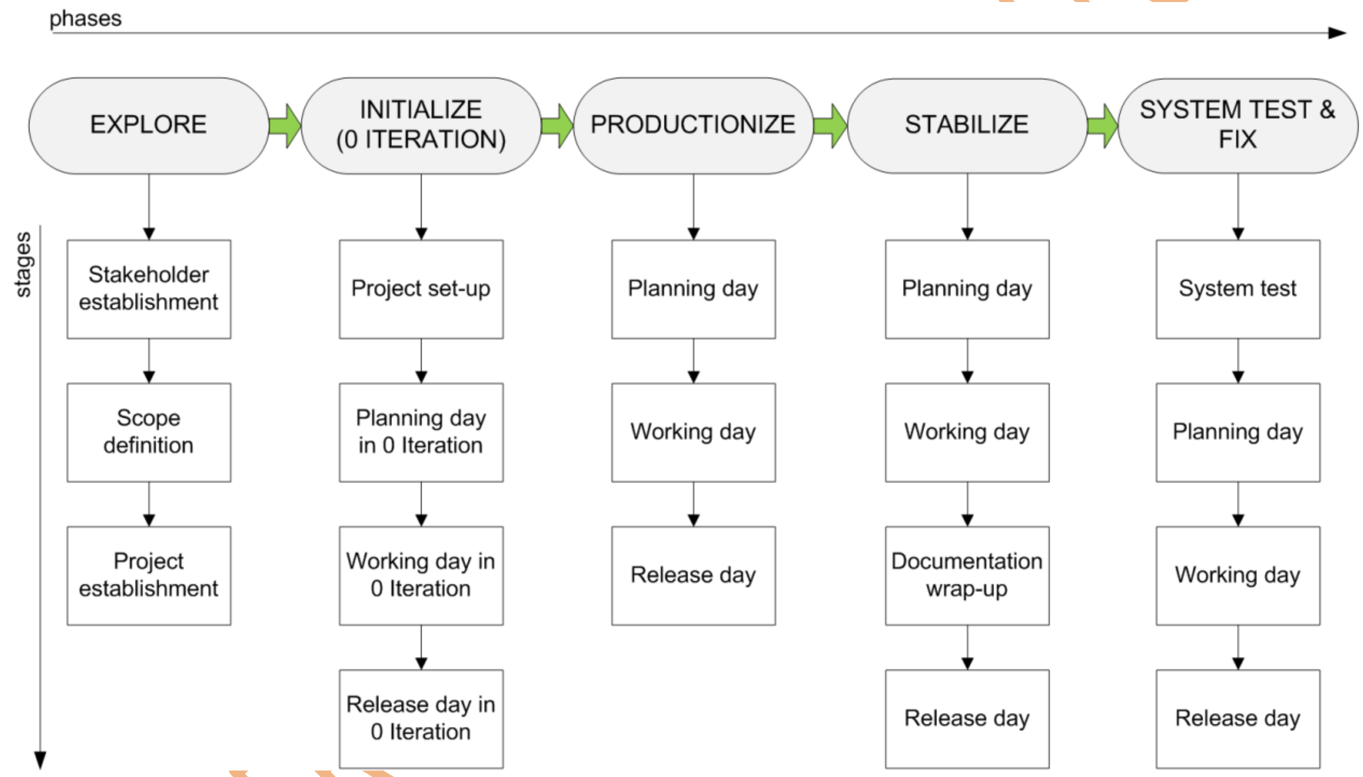
This method of Mobile D apps development consists of five phases namely; Explore, Initialize, Productionize, Stabilize, and System Test and Fix. Each of these phases has a number of associated stages, tasks and practices. The first phase involves the creation of a plan and the establishment of the project characteristics. The Explore phase is further split into three stages namely; stakeholder establishment, scope definition and project establishment. Under this phase, the main tasks that are solved include: identifying customers that will take part the development, initial project planning and requirements collection, and process establishment.

In initialize phase also consist of three stages namely; project set-up, initial planning and trial day. At this stage, all the team members analyze and understand the development process, after which they prepare the fundamental resources (physical, technological, and communications resources) that will be required for the production activities.

The Productionize phase is mainly concerned with the implementation activities. This phase is split into planning days, working days and release days. During the planning days, the concentration of the team members is mainly on enhancing the development process, prioritizing and analyzing requirements, planning the iteration contents and creating acceptance tests. During the working days, the functionalities of the software are implemented with the help of a Test-Driven Development (TDD) practice. It's also at this stage that the developers create unit tests. This is usually created by using TDD and Continuous Integration. The written codes must pass the unit test. If the code fails the test, then a new code will be created and integrated into the existing version of the product. Finally, a working version of the system is produced and validated through acceptance testing, at the release days.

The Stabilize phase is for finalization of the products, while the System Test & Fix phase is for testing product. These two phases consists of stages that are quite similar to those of Productionize phase. The only different is that they are modified to accommodate documentation building and system testing

Figure 4: The Mobile D process



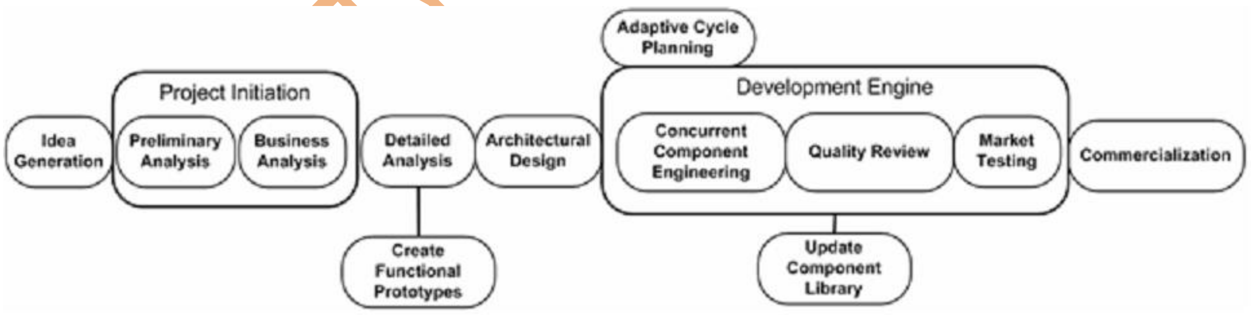
The Mobile-D process has been successfully applied in the development of Mobile Software applications. It has helped to boost the visibility of the application, as well as the early discovery and repair of technical errors. These always result to the production of apps, with low defect density.

Hybrid Method Engineering

The Hybrid Method Engineering is suitable mobile application development process that employs a methodological engineering approach (Rahimian & Ramsin, 2008). It is mainly based on such practices like agility, review meeting, reusability support, market awareness, market base architecture, etc. Thu, it's a unique approach that can be adopted in risk-based methodologies for mobile applications. The Hybrid Method Engineering also makes use of pre-established key requirements and conclusions as input, to iteratively generate the desired methodology. The mobile development methodology proposed by Rahimian & Ramsin (2008) consists of four iterations, which are generated from the generic software development lifecycle (Analysis, Design, Implementation, Test, and Transition).

In the first iteration, the methodology is detailed by including those practices associated with agile methods. The second iteration is concerns with all issues that are normally encountered, when introducing new products and services to the market. In other words, it involves, giving special attention to the market situations. Practically, the second iteration involved the conduction of market analysis and subsequent integration of activities associated with New Product Development. This will help the developer to gain a competitive advantage over its rivals.

Figure: The Hybrid Method Engineering



Source: Rahimian & Ramsin; 2008

The third iteration, involves the integration of Adaptive Software Development (ASD) ideas into the methodology, while the final iteration is all about adding prototyping, with the sole aim of mitigating likely technology-related risks.

In summary, this unique methodological approach begins with the identification of specific requirements for a mobile software development methodology. The identification process is determined by the characteristics expected on activities and intermediate products through the development process. Structurally, the baseline for the Hybrid Methodology Engineering Process is based on generic software development cycle, which is customized by merging the general Agile practices with principles of Methodology Engineering for new product introduction. Thus, the overall approach that is subsequently obtained, cover from the generation of the idea up to the product release. Hybrid Methodology Design defines a of a top-down, iterative-incremental process that considers the Prioritization of the Requirements, an Iterative Development Engine using traditional principles of Extreme Programming, such as Test-Driven Development and continuous integration. Market testing and commercialization activities take place afterward, in order to prepare the product to be profitable. The Hybrid Method Engineering is high level and no case studies or test result can be found from literature studies (Kaleel et al, 2013).

The impacts of Mobile D Process and Hybrid Method Engineering on apps development

One of the biggest challenges facing companies that specialize in app development is the ever changing taste of apps users. This shortens the average lifespan of a mobile app to approximately 12 months. Thus, it's very essential for commercial app developers to have the ability to quickly develop software solutions. Through this way, companies that develop mobile apps will be able to bring relevant and functional applications to the market. Coincidentally, the principles of Mobile D and Hybrid Method Engineering establish unique frameworks that allow the developers to develop and release mobile apps so they have the longest life span in the marketplace. This is usually accomplished by consistent release of recurring versions of the mobile applications.

Normally, a lot of time is required for mobile app developers to write and test working software. Unfortunately, it's not practical for developers to spent most of their time in writing documentation. Consequently, there is always the need to have a technological tool, whose principles align well with the requirement for mobile software development projects. This is exactly one effect of Mobile D process and Hybrid Method Engineering. The adoption of these two methodological approaches has resulted to rapid delivering of mobile applications that have high adaptability to various mobile operating platforms. Thus, the development and subsequent release of mobile software no longer requires much time duration.

The adoption of Mobile D process and Hybrid Method Engineering has enabled mobile apps companies to execute lightweight projects with continuous planning as against the provision of detailed projects with long term planning, which used to be the case in previous cases. The consistent delivery of mobile applications in sprints enables the software to be produced for clients on regular basis.

A case study of Accenture

Accenture is a global management consulting, technology services and outsourcing company, with more than 246,000 people serving clients in more than 120 countries. In addition to other sectors, Accenture also specializes in Mobile Services, where its main focus is to help clients accomplish breakthrough growth in the rapidly changing mobile ecosystem. The main mobility services offered by the company include: consulting, software services – applications, software services – devices and platforms, managed services, and business integration services. It also provides mobile application across a wide range of platforms and industries such as Windows, Android, Blackberry, Apple iOS, Symbian, Linux and Meego. The Agile Method of Software developments such as Mobile D Process and Hybrid Method Engineering are fully employed in rendering mobile services to clients. The adoption of this methodological approach, has greatly changed the pattern, the company follows in rendering services to its immense clients (Gartside, et al, 2013).

In other to understand how Agile methods like Mobile D Process and Hybrid Method Engineering have changed the pattern of Accenture Mobile Services, we focused our case study on how the company created an Android App as well as the changes that have taken place in tits Human Resource unit. In the first scenario, the main objective is to create an app for tracking expenses for highly mobile employees, which will enable them to correctly compile and submit their travel costs, without much ado. The procedure employed in the development of the software is as follows (Accenture, 2013):

- **Initial requirements capture and discussion:** At this preliminary stage, the scope of the android apps and how it will access data sources are determined.
- **Initial code prototyping.** After the identification of the scope of the apps, the next step was the creation of a prototype of the appropriate code. This accommodates front-end, back-end, and connectivity layers.
- **Interface design:** At this stage, the Android user interface (UI) framework was use to designed the app's interface.
- **App implementation:** Here, all the developers involved in the development of the apps, came together and present their own work to generate the initial version of the app. Through this way, the developers ensured that all the pieces of code communicate properly.
- **Quality Assurance (QA):** The developers established extensive QA techniques, which are specifically aimed at ensuring the obtainment of high quality android apps. The main QA techniques that are used in the development include: automated UI testing, built-in automated remote crash reporting and testing against replicated back-end server.
- **Security review:** This involves the application of strong and tested encryption capabilities to the functional and structural design of the android apps. The essence is to ensure security, leveraging the OpenSSL security stack that is available on the platform.
- **Beta trials:** At this stage, the beta version of the android apps was made available to a selected user group. This enabled the developers to tracked defects and usability issues and recorded them in an internal defect tracking tool. This will helps in proper management of the life cycles of the android applications.
- **Upload to Google Play:** It's at this stage that the android apps are made available on the Google Play.
- **Feedback mechanism:** After the android apps was made available, Accenture developers created two types of feedback mechanisms. One track stability issues for the application, while the second mechanism enables users to indicate the features they would like developers to include into the functional structure of the android apps, in the future. In the first mechanism, the identified issues can be transmitted back to the Accenture IT, only if the user allows it.

The practical application of the agile methodologies has greatly improved the functional capability of Accenture. Unlike in previous cases, the process of software development can now been done in a quicker and more innovative way. This is clearly evident in development of the android application that has already been discussed above. Such functionality has also made it possible for Accenture developers to execute their strategy faster as it used to be prior to the adoption of Agile methods. Finally, the adoption of the Agile methodologies has revolutionalize the nature of mobile applications, which are normally provided by Accenture. For instance, the mobile consumer applications provided by the company has now changed from the simple ringtones and wallpaper to more sophisticated apps like gaming, social networking and mobile commerce, including banking, payments, coupons and ticketing.

CONCLUSION

This paper extensively analyzes the successful use of Agile approach in software development. Some of the Agile engineering paradigms are product creation frameworks, which follows project management practices that are specifically aimed at involving customers in the development process and delivering iteratively. These give developers the highly desired flexibility, required in service delivery. Thus, Agile methodologies are exclusively suitable for developing mobile applications as such projects are normally required to be executed within short time. However, it's very important for mobile app developers to always consider the different Agile approaches, during the execution of a project. Through this way, the traditional software development process can be greatly improved. This will also help in the identification of other types of software development, where Agile may be beneficial to conducting software process improvements. In conclusion, this study shows that the speed and quality of mobile application development can be greatly enhanced through the adoption and adapting of these Agile methods.

BIBLIOGRAPHY

Accenture (2013). Android Application: Case Study Simple App, Sophisticated Effort. http://www.accenture.com/SiteCollectionDocuments/PDF/Accenture-Future-of-HR-Trends-Agile-Organizations.pdf Retrieved 27th July, 2014.

Ahmed, A., Ahmad, S., Ehsan, N., Mirza, E., and Sarwar S., (2010) “Agile Software Development:Impact on Productivity and Quality” , pp. 287-290, IEEE 2010

Begel, A. and Nagappan, N (2007) “Usage and Perceptions of Agile Software Development in an Industrial Context:An Exploratory Study”, First International symposium on empirical software engineering and measurement, pp. 255-264, 2007.

Dingsøyr, T., Nerur, S. P., Balijepally, V. & Moe, N. B. (2012). A decade of agile methodologies: Towards explaining agile software development.. *Journal of Systems and Software*, 85, 1213-1221.

Dyba, T., & Dingsoyr, T. (2009). What do we know about agile software development?. *Software, IEEE*, *26*(5), 6-9.

Flora H. K., Chande, S.V. and Wang X (2014). Adopting an Agile Approach for the Development of Mobile Applications. *International Journal of Computer Applications (0975 – 8887) Volume 94 – No.17, May 2014 pp 43-50*

Gartside, D., Gossage, W., Silverstone, Y., Tambe, H. and Cantrell, M. (2013). Trends Reshaping the Future of HR: HR Drives the Agile Organization. http://www.accenture.com/SiteCollectionDocuments/PDF/Accenture-Future-of-HR-Trends-Agile-Organizations.pdf Retrieved 27th July, 2014.

Ho, J. (2008). Issues and Challenges of Agile Software Development with Scrum. *Issues in Information Systems*, *9*(2), 188-195.

Kaleel, Shakira Banu and Harishankar, Ssowjanya (2013), "Applying Agile Methodology in Mobile Software Engineering: Android Application Development and its Challenges". *Computer Science Technical Reports*. Paper 4. http://digitalcommons.ryerson.ca/compsci_techrpts/4

Khambati, A., Grundy, J., Warren, J., & Hosking, J. (2008). Model-driven Development of Mobile Personal Health Care Applications. *Proceedings of the 23rd IEEE/ACM International Conference on Automated Software Engineering*, (pp. 467-470)

Kim, H., Choi, B., & Wong, W. E. (2009). Performance Testing of Mobile Applications at the Unit Test Level. *Proceedings of the 2009 Third IEEE International Conference on Secure Software Integration and Reliability Improvement*, (pp. 171-180).

Klimeš, C. and Procházka, J. (2006) New Approaches in Software Development. In Acta Electrotechnica et TInformatica, 6(2), 2006.

Kumar, G. and Bhatia, P.K. (2012). Impact of Agile Methodology on Software Development Process. International Journal of Computer Technology and Electronics Engineering (IJCTEE) Volume 2, Issue 4, August 2012. ISSN 2249-6343

Leffingwell, D. (2007). *Scaling software agility: best practices for large enterprises*. Addison-Wesley Professional

Lemétayer, J. (2010). identifying the critical factors in software development methodology FIT.

Livermore, J.A (2007). "Factors that impact implementing an Agile Software Development Methodology", pp. 82-85, IEEE 2007.

Maher, P. (2009) "Weaving Agile Software Development Techniques into a Traditional Computer Science Curriculum", Proc. of 6th IEEE International Conference on Information Technology: New Generation, pp. 1687-1688, 2009.

Maurer, F. and Martel, S. (2002) Extreme programming. Rapid development for Web-based applications. In Internet Computing, IEEE, 6(1), 2002, 86-90.

Mordinyi, R., Kuhn, E and Schatten A (2010), "Towards an Architectural Framework for Agile Software Development", 17th IEEE International Conference and workshops on Engineering of Computer Based Systems, pp. 276- 280, 2010.

Nakki, P, Koskela, K and Pikkarainen, (2011) M."Practical model for user-driven innovation in agile software development", Proc. Of 17th International Conference on Concurrent Enterprising, pp. 1-8, 2011.

Nerur, S., Mahapatra, R., & Mangalaraj, G. (2005). Challenges of migrating to agile methodologies. *Communications of the ACM*, *48*(5), 72-78.

Osama Sohaib, Khalid Khan, (2010)"Integrating Usability Engineering and Agile Software Development: A Literature Review", International Conference On Computer Design And Appliations (ICCDA 2010), Vol. 2, pp. 32-38, IEEE 2010.

Rahimian, V. Ramsin, R. (2008) "Designing an Agile methodology for mobile software development: A hybrid method engineering approach", in proceedings of the Second International Conference on Research Challenges in Information Science

Salo, O., & Abrahamsson, P. (2008). Agile methods in European embedded software development organisations: a survey on the actual use and usefulness of Extreme Programming and Scrum. *Software, IET*, *2*(1), 58-64.

Vladan Devedzic and Sasa R. Milenkovic, (2011) "Teaching Agile Software Development: A Case Study", IEEE transactions on education, vol. 54, no. 2, pp. 273-278, May 2011.

Wang, X., (2011) "The Combination of Agile and Lean in Software Development: An Experience Report Analysis", IEEE Agile Conference, pp. 1-9, 2011.

Wang, Y, Sang, D, and Xie, W, (2009). "Analysis on Agile Software Development Methods from the View of Informationalization Supply Chain Management", 3rd International Symposium on Intelligent Information Technology Application Workshops", pp. 219-222, 2009.

West, D., Grant, T., Gerush, M., & D'Silva, D. (2010). Agile development: Mainstream adoption has changed agility. *Forrester Research*.

Zuo, A, Yang, J. and Chen, X (2010) "Research of Agile Software Development Based on Formal Methods", International Conference on Multimedia Information Networking and Security, pp. 762-766, 2010