Empirical Investigation of Agile Software Development: A Cloud Perspective

Anupriya Tuli
Amity University Uttar Pradesh
Noida, India
anupriyatuli6@gmail.com

Nitasha Hasteer
Amity University Uttar Pradesh
Noida, India
nhaster@amity.edu

Megha Sharma
Amity University Uttar Pradesh
Noida, India
megha.rsystems@gmail.com

Abhay Bansal
Amity University Uttar Pradesh
Noida, India
abansal1@amity.edu

ABSTRACT

Technological advancements have contributed to more complex software demands. The Agile approach to software development is widely practiced by the software development industry as it offers faster production with a promise of better software quality. It also provides a flexible process to accommodate changes during the software development life cycle, as per dynamic user requirements. In this paper we characterize the significance of software development models and the role of Agile methodologies in today's dynamic world of technologies. The purpose of this paper is to empirically investigate the choice between two of the Agile methodologies, Scrum and Extreme Programming. The outcome of the investigation based on secondary sources and limited primary sources, reveals that current software industry practices tend to opt for Scrum-based development. This work highlights the benefits of bringing the cloud and Agile methods of software development together, to fully realize the potential of the distributed paradigm.

Keywords

Software Development Life Cycle (SDLC), Agile Methods, Scrum, Extreme Programming (XP), Cloud Computing

1. INTRODUCTION

The software development approach plays an important role in the quality of software being developed. A non-systematic development approach will produce low-quality software that may not meet user expectations and also has a risk of incurring additional cost of development. According to Sommerville [2] a software development model provides a high-level abstraction of a set of best software-development practices.

The software development process has progressed from the traditional waterfall model to Agile processes with the shift from static to dynamic industry demands and ever-advancing technologies. Agile processes unlike the traditional models focus on incorporating change within software during the process of software development as they have shorter development cycles and higher levels of client involvement.

In the present scenario of electronically-distributed software systems, the concept of software-as-a-service (SaaS) has gained popularity. Agile processes have a strong focus on collaboration among developers and customers via feedback loops. The fact that the cloud is seen as a business opportunity and platform to promote collaboration makes it a more attractive option to implement Agile.

This work provides a comparison between two of the most-widely accepted Agile methods, Extreme Programming and Scrum-based development. It would enable the practitioners of Agile to choose between these approaches. This paper also explores the idea of combining the power of Agile and cloud. The rest of this paper is organized as follows. Section 2 includes a literature review, followed by section 3, which outlines the significance of software development models. In Section 4, we throw light on Agile methodologies and their practices. In Section 5, we analyze two widely-accepted Agile models Scrum and Extreme Programming. In Section 6 we discuss using the cloud for Agile development, leading to conclusions in Section 7.

2. LITERATURE REVIEW

The early research on Agile focused on issues related to the adoption of Agile methods. Less work in the literature investigates the amalgamation of Agile with cloud. Andrea Magdaleno et al. [6] presented a quasi-systematic review of reconciling software development models. They found that there exist distinct organization, group and process levels of reconciliation amongst software development models.

The Standish Group's 2011 CHAOS report states that Agile methods are three times as successful as the traditional approaches. In their work, Felipe Carvalho et al. [21] outlined the fact that though all the techniques that constitute the Agile group have overlapping subsets of the features of the Agile manifesto [20], still they vary in their relevant characteristics.

The work done by O. Salo and P. Abrahamsson [9] demonstrates a clear trend towards higher adoption of Extreme Programming practices in comparison to Scrum in embedded software development organizations across Europe. Another empirical study conducted at Microsoft by Begel, A. and Nagappan [4] revealed that one-third of the study respondents used Agile methodologies. This study shows that Scrum is the most popular agile methodology practiced at Microsoft. Asif Qumer et al. [26] presents a comparative analysis of Extreme Programming and Scrum, using the 4-Dimensional Analytical Tool (4-DAT). This comparison is based on scope, agility, Agile values and software process. Their work highlights that Scrum supports more Agile practices, but has less Agile phases than Extreme Programming.

Software as a Service (Saas) has gained popularity in recent times along with distributed software development systems. A team of researchers at Collabnet [24] is working on a development platform in the cloud to simplify the Agile issues in a cost effective way. The work of Amit Dumbre et al. [14] demonstrates that Agile software development practices are significantly enhanced with cloud technologies and platform as a services such as Azure Platform as a service, to provide an on-demand, scalable environment to achieve faster software delivery.

3. SOFTWARE DEVELOPMENT MODELS

Software engineers aim to use software development models for building fault free software that meets user requirements and is delivered within the specified time limit and budget. The use of software development models enables organized development of software. Without these models, software development teams may produce low-quality software, which is difficult to maintain and violates the budget and time deadlines.

In 1968, during the first NATO Software Engineering Conference, the term Software Crisis was coined to refer to the difficulty of writing correct, understandable and verifiable software [18]. The convergence of communication, computer science and GUI encouraged the adoption of planned software development methods. After 1968, technical advancements have made software development more complex in nature, eliminating informal approaches of software development, which were earlier being used for developing simple software. This transformation gave birth to a situation where hardware cost was decreasing and software cost was increasing. Developing software using software development models gave a solution that was required in the software industry to control the software crisis [16].

Software development models can be categorized into plan-driven, Agile and FOSS (Free and Open-Source Software) development models. These models follow different approaches, but have the same goal to enhance the end product by using a systematic software development process. Agile development models help in building flexible software and enable teams to respond to the unpredictability of software development. The plan-driven models seek reliability, predictability and stability. The FOSS development models develop software products based on special characteristics. Among these software development models we find similarities as well as many differences. Plan-driven, Agile and FOSS development models follow different approaches to software development [6].

4. AGILE SOFTWARE DEVELOPMENT

Agile can be defined as the software development methods where tasks are carried out in small work phases, characterized by retrospective analysis, contributing to frequent modification to the software in accordance with the ever-changing requirements of customers. The Agile processes have a group of software development methods to execute iterative and incremental development, by replacing high-level design with frequent redesign.

With rapid technological enhancements, the software being constructed today is becoming more and more complex. To meet the dynamic demands of customers and avoid a software crisis, companies are adopting Agile methods. A Survey conducted by Murphy et al. [22] shows an increase in Agile methods being practiced within Microsoft from 2006 to 2012.

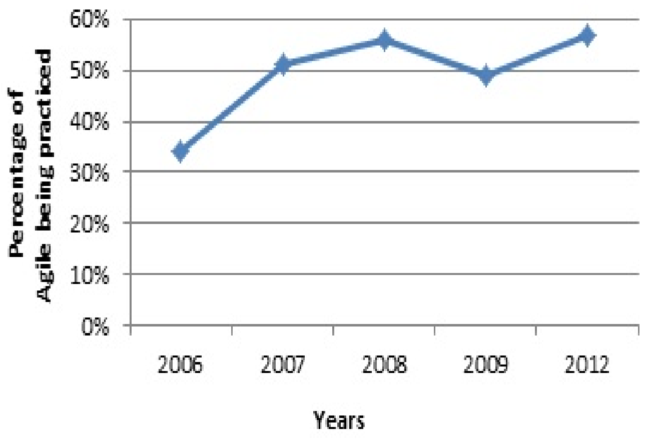

Figure 1. Percentage of Agile methods practiced in Microsoft over the years

The Agile umbrella provides shelter to many software development methodologies including Scrum, Extreme Programming, Crystal, Kanban, FDD (Feature-driven development), DSDM (Dynamic Systems Development Method), etc.

4.1 Extreme Programming (XP)

Sommerville [17] states that Extreme Programming is a planned and iterative method to develop software with extreme levels of involvement by customers in the construction and testing phases. The features of XP that are uncommon to the rest of its counter approaches are [17, 19]:

4.1.1 Requirements

These are expressed as scenarios by users, which are documented in the form of Story Cards. The development team breaks each story card into a series of small tasks, which are prioritized by the customer for development.

4.1.2 Planning Game

The development team estimates the effort needed for implementation of Story Cards. It is on the basis of this estimation that the customer decides the scope and release time of the tasks.

4.1.3 Simple Design

XP does not accommodate changes in the design, which causes degradation of the software structure. The developers nullify this software structure degradation by performing refactoring. In refactoring the software is immediately updated whenever there is a scope for improvement.

4.1.4 Pair Programming

Programmers work in dynamic pairs encouraging communication, group ownership and reduction of workload and working hours.

4.1.5 Test First Development

This is a unique concept where a test case is written prior to coding for the task. XP is a software development model with testing being an integral part.

4.1.5.1 The customer is responsible for developing acceptance test cases and validations.

4.1.5.2 Acceptance testing is incremental in nature as series of test cases are executed.

4.1.5.3 Testing is done using automated test harness tools.

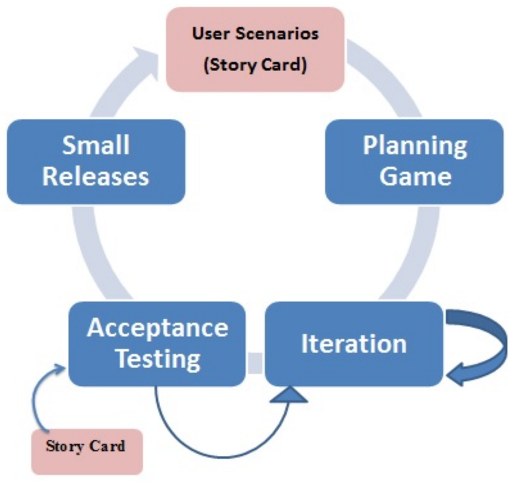

Figure 2. Extreme Programming approach

The French Scrum User Group [12], which is an affiliate of the Scrum Alliance, conducted a national survey on Agile methods in France in June 2009. The survey, with the participation of 230 industry professionals from over 150 companies ranging from E-commerce, banking, insurance, scientific innovation, software development and IT service providers, showed that 52% of companies were using XP. Andrew Begel et al. [15] conducted a survey at Microsoft Corporation to study the perception towards pair programming, one of the engineering practices advocated by XP. Their survey indicates that 64.4% of respondents believe that pair programming works well for them and produces higher quality code. O. Salo and P. Abrahamsson [9] conducted a survey on the use of Extreme Programming in embedded software development organizations across Europe. They found that 54% of the participating organizations applied XP.

4.2 Scrum Methodology

In the last decade of software development, there has been challenges related to methodologies and best practices to deliver software. People used to do work with the help of defined roles, defined models, and defined practices to execute defined plans. In the present times, there is a need for an approach through which resource, schedule and cost constraints could be best managed. One of the methodologies currently used is Scrum. It is a lightweight methodology that uses fixed-length iterations known as sprints. A sprint includes all aspects of work, i.e. designing, coding, testing, etc. Scrum methodology has the following roles:

4.2.1 Product Owner

The product owner is responsible for product vision, maximizing return on investment, prioritizing the product backlog, which is the list of customer-centric features and also deciding whether to continue development or not.

4.2.2 Scrum Development Team

The team is responsible for doing each and every task as assigned by the product owner within the sprint deadline. A team of 5 -7 members is made to carry out the development tasks.

4.2.3 ScrumMaster

The ScrumMaster acts as a facilitator who resolves obstructions and also helps in creating a self-organizing environment.

Further, there are Scrum meetings, which are initiated by the ScrumMaster, although the ScrumMaster has no decision-making authority at these meetings. In the Sprint planning meetings, the Product Owner assigns a priority to the tasks from the product backlog items (created by Product Owner). Selected work is then shifted from the product backlog to the sprint backlog. Team members decide a particular place and time within the office premises where they discuss their work on a daily basis. In a Sprint review meeting after completion of each Sprint, the team gives a live demonstration of the product increment to the Product Owner. The Product Owner then decides whether to accept the task as complete or not. Lastly, in Sprint retrospective meetings, all the team members review their work completed during a particular Sprint. They conclude whether to continue in the same manner or adopt a different strategy.

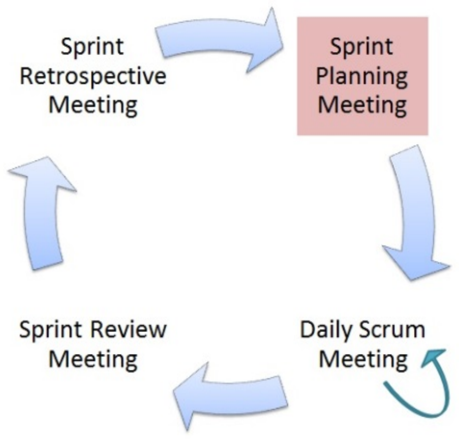

Figure 3. Scrum meeting flow

A survey conducted by the French Scrum User Group [12] in 2009 showed that 86% of the companies were using Scrum. A Danube Case study by Pat Elwer [13] discussed the adoption of Scrum by Intel in 2008 for one of its products. The outcome of the project showed a 66% reduced cycle time, performance to schedule, improved morale of employees, and increased transparency. According to this report, Scrum was considered as the best project management framework to employ. Another survey on the use and usefulness of Scrum in embedded software development organizations across Europe by O. Salo and P. Abrahamsson [9] showed that 27% of the participating organizations applied Scrum. However, there are a few challenges that organizations may encounter while adopting the Scrum methodology:

- Cost to company may increase due to staff turnover.
- Availability of skilled resources.
- Ability to build applications from multiple locations in projects with geographically distributed teams.
- Test-driven development is difficult as it sometimes leads to excess code generation.

5. COMPARISON BETWEEN SCRUM AND EXTREME PROGRAMMING

The Agile manifesto [20] states a set of principles, which are practiced using different software development models collectively known as Agile processes. Although all of these development models are under the Agile umbrella, they vary on different parameters of Agile principles. Empirical study of both Scrum and XP variants of the Agile methodology led to the formulation of the comparison presented in Table 1.

Table 1. Comparison between XP and Scrum

Parameter	**Scrum-Based Development**	**Extreme Programming (XP)**
Relevant Characteristic [21]	Lifecycle management	Software construction
Customer Involvement [10]	Not specifically onsite	Onsite customer
Requirement Documents	Managed Requirements in form of ARTIFACTS: Product backlog, Sprint backlog	Managed Requirements in form of STORY CARDS.
Project Management Process [26,10]	Sprint planning meeting, daily Scrum meeting and Sprint review	Planning game
Facilitator [10]	ScrumMaster	XP coach
Module Development Priority [17]	Priority by Scrum product owner	Priority by customer
Design Principle [26]	Complex Design and believes in design for change	Simplification of code and unanticipated change implementation (refactoring)
Development Approach	Iterative (Sprint) + Incremental	Iterative + Incremental
Coding	Coding Standards are not used	Coding Standards are used
Working Hours [17]	Programmers have long working hours	Programmers work in pairs which reduces workload and working hours (pair programming)
Testing	No formal testing approach	Test first approach, automated and incremental acceptance testing
Product Ownership [10]	Product Owner is responsible for the product	Group responsibility

Quick interviews were conducted with the representatives of nine software development companies. It was found that only one out of the nine companies that participated in the interview currently implements XP, whereas the rest of them practice Scrum.

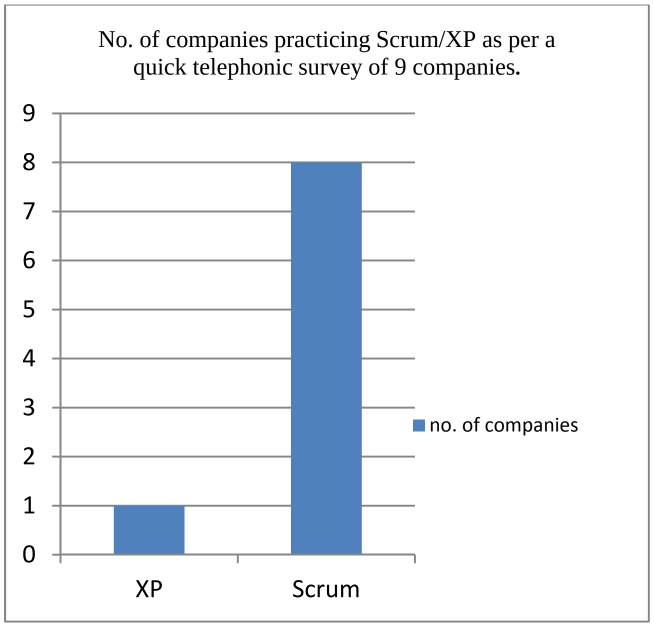

Figure 4. Scrum is being practiced more as compared to XP

Table 2. Companies practicing XP/Scrum

Agile Method	**Company Name**
XP	• Vismaad (http://vismaad.com/software.htm)
Scrum	• Computer Sciences Corporation (http://www.csc.com/in) • Toshiba Software Pvt. Ltd. (http://www.toshiba-tsip.com/tsip/) • Mercer (http://www.mercer.co.in/home) • Accenture Services Pvt. Ltd. (www.accenture.com) • Avalon Technologies, Inc. (avalontech.net) • Kellton Tech (www.kelltontech.com) • Deloitte Touche Tohmatsu Limited (https://www.deloitte.com/in)

6. CLOUD FOR AGILE DEVELOPMENT

The usage of Agile software development is increasing due to many benefits like increased productivity, better quality, more flexibility and collaborative processes to accommodate changes. But there can be many unanticipated issues like scalability that can arise in parallel with the use of Agile software development. Also, providing and managing development tools and infrastructure that support the Agile process could be challenging for enterprises. Using Cloud Computing would help Agile enterprises in this regard. By leveraging a cloud development platform, an enterprise can start using development tools that support the Agile process, cost effectively, without provisioning and managing the development infrastructure [24].

Basically Agile depends on many different tools to support its practices like feedback, transparency and prioritizing tasks. An Agile organization requires development infrastructure and an integrated toolset. All these requirements are fulfilled more easily with the help of a cloud development platform. The following are few of the benefits of bringing cloud and Agile together [14, 24]:

6.1 Distributed Application Development

The cloud supports distributed application development with teams dispersed geographically. Any team member can have access to information about the application at any time and at any place.

6.2 Data Sharing

Resources are required to share a huge amount of data between different machines for many applications. The cloud plays a crucial role by providing a medium to share large amounts of data frequently as per the requirements.

6.3 Prioritizing Tasks

Continuous prioritizing and reprioritizing the tasks within a sprint needs tracking and transparency. In a distributed system, there is a requirement of a tool that can be accessed by all team members to keep them updated. Deploying such a tool internally consumes energy and takes time. With a cloud development platform, organizations can provision Agile Management tools in minutes, and can scale globally with thousands of users.

6.4 Transparency

Transparency comes through data capture. In a cloud-based Agile development, organizations can have developer services that capture and share data from all distributed code repositories, build and test environments. By capturing the needed and important data, organizations can measure performance, observe, supervise and manage the Agile project.

6.5 Infrastructure

Agile software development requires Infrastructure to be in place from the start of the project. Cloud helps by providing infrastructure from the first day of the project. Additional overhead of software and hardware upgrade is taken care of, by the Cloud provider.

6.6 Benefits Business Users

Cloud facilitates Agile practices in order to meet the users requirements faster. The cloud supports the concept of “pay per use,” easy deployment of applications to Go Live and also provides faster “time to market” services and products.

7. CONCLUSION

Agile philosophy is an answer to many important issues of new software reality ranging from distributed software development to software as a service (SaaS). The increase in adoption of Agile practices with time has shown the promising aspect of this approach.

Agile brings in added value and ensures better returns for developing software and services on cloud. The salient feature of the Agile approach is collaboration between customers and organizations. The power of Agile can increase with Cloud-based development. The idea of the amalgamation of Agile and cloud not only solves the scalability issues, but also augments existing Agile practices to deliver software faster.

8. REFERENCES

[1] Da Silva, E.A.N., Lucredio, D. 2012. Software Engineering for the Cloud: A Research Roadmap. In

Software Engineering (SBES), (2012) 26th Brazilian Symposium on, 71- 80. DOI= 10.1109/SBES.2012.12.

[2] Sommerville, I. 2004. Software Engineering, Addison Wesley, 7 Ed.

[3] Highsmith, J. and Cockburn, A. 2001. Agile Software Development: The Business of Innovation. *Computer*, vol. 34, 120-122. DOI= 10.1109/2.947100

[4] Begel, A. and Nagappan, N. 2007. Usage and perceptions of Agile software development in an industrial context: An exploratory study. In *Empirical Software Engineering and Measurement*, (Washington, 2007), 255 - 264. DOI= 10.1109/ESEM.2007.12.

[5] Yau, s.s., and An,H.G. 2011.Software Engineering Meets Services and Cloud Computing. *Computer* ,Vol 44, no 10 (Oct. 2011). DOI= 10.1109/MC.2011.267

[6] Magdaleno, A.M., et al., Reconciling software development models: A quasi-systematic review, Journal of Systems and Software, v.85 no.2, 351-369 (February, 2012).DOI=10.1016/j.jss.2011.08.028

[7] Rising, Linda, Janoff, N.S. 2000. The Scrum Software Development Process for Small Teams, *IEEE Software*, vol. 17, no. 4 (July/Aug. 2000), 26-32. DOI= 10.1109/52.854065.

[8] Williams, L., Cockburn, A. 2003. Agile software development: it's about feedback and change, Computer, Vol.36, no.6 (2003), 39 – 43. DOI= 10.1109/MC.2003.1204373.

[9] Salo, O. and Abrahamsson, P. 2008. Agile Methods in European Embedded Development Organizations: a survey study of Extreme Programming and Scrum, *IET Software*, vol. 2 (February, 2008), 58-64. DOI = 10.1049/iet-sen:20070038.

[10] MSDN Library, Agile Principles and Values by Jeff Sutherland, *Microsoft Developer Network White Paper*. http://msdn.microsoft. com/en-us/library/dd997578.as

[11] Standish Group 2011. The Crisis in Software: The Wrong Process Produces the Wrong Results, *The Standish Group Report* (2011). www.controlchaos.com/storage/S3D%20First%20Chapter.pdf.

[12] French Scrum User Group 2009. A National Survey on Agile Methods in France. (June 2009) www.frenchsug.org

[13] Elwer, P., et al.. 2008. Agile Project Development at Intel: A Scrum Odessey, Danube Case Study of Intel Corporation. http://Scrumtrainingseries.com/Intel-case-study.pdf

[14] Dumbre, A., et al. 2012. Practicing Agile software development on the Windows Azure Platform., White Paper Infosys (2012). www.infosys.com/cloud/.../practicing-Agile-software-development.pdf

[15] Begel, A., Nagappan, N. 2008. Pair programming: what's in it for me? , In Proceedings of the Second ACM-IEEE international symposium on Empirical software engineering and measurement, Kaiserslautern, Germany. DOI=10.1145/1414004.1414026

[16] Schach, S. 2007. Software Engineering, Tata McGraw Hill, Ed. 7, 4-6.

[17] Sommerville, I. 2005. Software Engineering, Pearson, Ed. 7, 26, 418 – 430.

[18] Randell, B. 1996. The 1968/69 NATO Software Engineering Reports.http://homepages.cs.ncl.ac.uk/brian.randell/NATO/NATOReports.

[19] Beck, K. 1999. Embracing Change with Extreme Programming. *Computer*, vol.32, 70-77. DOI=10.1109/2.796139

[20] Beck, K., et al. 2001., Manifesto for Agile Software Development. http://Agilemanifesto.org/

[21] Carvalho, F., et al.. 2013. Service Agile Development Using XP, SOSE, IEEE Conference Publication, 254 – 259.DOI=10.1109/SOSE.2013.25

[22] Murphy, B., et al.. 2013. Have Agile Techniques been the Silver Bullet for Software Development at Microsoft?, Empirical Software Engineering and Measurement, ACM / IEEE International Symposium,(2013), 75-84. DOI=10.1109/ESEM.2013.21

[23] Hasteer, N., Bansal, A., Murthy, B.K. 2013. Pragmatic Assessment of Research Intensive Areas in Cloud : A Systematic Review, In *ACM SIGSOFT Software Engineering Notes*, Vol. 38, no.3 (May 2013), ACM New York, NY, USA, 1-6.

[24] Willie, 2011. Reinforcing Agile Software Development in the Cloud, *White Paper CollabNet Inc*. https://www.open.collab.net/media/pdfs/CollabNet%20Whitepaper_Reinforcing%20Agile%20Dev%20in%20the%20Cloud.pdf?_=d

[25] Munshi, D. 2009. Adopting Agile Methodology in a Large Scale Distributed Development Environment, *TCS White Papers* (2009).http://www.tcs.com/sitecollectiondocuments/white%20papers/GCP_white-paper_Agile_Methodology_Large_Scale_Distributed_Development_Environment_01_2010.pdf

[26] Qumer, A., Henderson-Sellers, B. 2006. Evaluation of XP and Scrum using the 4D analytical tool (4-DAT), European and Mediterranean Conference on Information Systems (July 6-7 2006).epress.lib.uts.edu.au/research/bitstream/handle/.../2006005499.pdf?

[27] Ramanujam, R., et al.. 2011. Collaborative and Competitive Strategies for Agile Scrum Development. In proceedings of 7th International conference on Network Computing and Advanced Information Management (*NMC), IEEE Conference Publication*, 123 – 127.