PROCEEDINGS OF 3RD INTERNATIONAL CONFERENCE ON COMPUTING AND INFORMATICS

8-9 JUNE 2011
BANDUNG, INDONESIA

AGILE SOFTWARE DEVELOPMENT PRACTICES THAT INFLUENCE SOFTWARE QUALITY : A REVIEW

SHAFINAH FARVIN PACKEER MOHAMED
FAUZIAH BAHAROM
AZIZ DERAMAN
JAMAIAH YAHYA

AGILE SOFTWARE DEVELOPMENT PRACTICES THAT INFLUENCE SOFTWARE QUALITY: A REVIEW

[1]Shafinah Farvin Packeer Mohamed, [2]Fauziah Baharom, [3]Aziz Deraman, [4]Jamaiah Yahya

Universiti Utara Malaysia, [1]shafinah@uum.edu.my, [2]fauziah@uum.edu.my, [4]jamaiah@uum.edu.my

Universiti Malaysia Terengganu, [3]nc@umt.edu.my

ABSTRACT. Agile software development (Agile) is being practiced in software industry nowadays as it fits the current business environment which focuses on delivering software to market as quickly as possible. In addition, Agile practitioners claim that it produces software with good quality. Thus, our research aims to identify Agile practices that should be followed in order to produce good quality software. Since many researchers report that the quality of people and process influence the quality of software product, this paper discusses on practices related to these two factors. The identified practices will be used for developing questionnaire in order to investigate current practice among Agile practitioners.

Keywords: Agile Software Development, Software Quality, Software Practice.

INTRODUCTION

Software quality has become a major strategic issue in software industry (Jamaiah, Fauziah, Aziz & Abdul Razak, 2005). This is because customers always expect that the software product, service and process to be good in quality (Lycett, Macredie, Patel & Paul, 2003), which meets their needs and follows certain standards (Krishnan, 1993). Besides, they also expect that software products can be developed faster (Verner, Liming, Babar & Ming, 2004). Generally, software with good quality has these criteria: 1) meets the expected requirements, 2) completed within budget, 3) completed on time, 4) completed in its entirety, 5) delivered together with a solid and thoroughly tested code, and 6) can be used easily (Nasution & Weistroffer, 2009). The quality of a software product highly depends on the people, organization and procedures used to create and deliver it (Fuggetta, 2000). However, according to Arthur (1993) and O'Regan (2002), there are three key elements need to be given attention in developing good software: 1) the quality of people involved, 2) the process performed and 3) the use of development technology. Nevertheless, Fauziah (2008) identified another two factors that influence software quality which are the working environment, and project condition. Besides, Hazzan and Dubinsky (2009) stated that human, technology used and organizational aspects should be considered in assuring software quality. Based on the literature mentioned, factors that influence software quality can be classified into five, which are process, human, working environment, technology and project conditon.

Realizing the needs for faster software development cycle and rapidly changing requirements, many organizations are shifting from conventional software development approach to Agile which is considered as a light-weight approach (Conn, 2004). However most Agile opponents argue that this software development approach is lack of documentation, which leads to software maintainability problem. In addition, Turk, France and Rumpe (2002) mentioned some limitations in Agile, such as, limited support for distributed development environments and limited support for building reusable artifacts. They believe these limitations able to give impact on the quality of developed software. On the other hand, many Agile representatives claim that it fits the industrial needs (Beck, 2000) because it promises higher quality software

(Sliger, 2006), higher customer satisfaction, lower defect rates, faster development times and becomes a solution to rapidly changing requirements (Boehm & Turner, 2003). Consequently, this paper reviews the literature related to practices of Agile development approach that must be performed in producing good quality software as part of our research work. Our research aims to construct a unified software process certification model which can be used to assess and certify software based the quality of development process no matter what approach was used either conventional, Agile or Web. However, discussion in this paper will focus on findings from literature about the required practices related to human and process factors of Agile based development. The structure of this paper is organized as follows: the next section discusses about Agile practices that influence software quality and finally future work and conclusion is provided.

AGILE SOFTWARE DEVELOPMENT PRACTICES THAT INFLUENCE SOFTWARE QUALITY

Agile is introduced recently as a consequence from the problems faced in conventional methodologies (Rico, Sayani & Sone, 2009) which are not flexible in accepting unstable and volatile requirements (Verner et al., 2004). It is aimed to produce higher quality software in a shorter period of time (Livermore, 2007; Sliger, 2006). Currently there are many Agile methodologies such as Extreme Programming (XP) and Scrum (Abrahamsson, Salo, Ronkainen & Warsta, 2002). These methodologies have similar values and practices, whereby they follow 12 principles, for instance: "Welcome changing requirements, even late in development" and "Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale" (Agile Manifesto, 2001). It also follows four values which are: 1) iterative, 2) incremental, 3) self-organizing and 4) emergent (Lindvall et al., 2002).

A survey conducted by Microsoft Researchers reveals the benefits of Agile as improved communication and flexibility, with faster release. Another survey conducted by IBM and Ambysoft shows that Agile adoption improves productivity, customer satisfaction and project success (Rico et al., 2009). The results of these studies show that Agile leads in producing good quality software. Therefore, there is a need to identify the Agile practices that influence software quality. Although factors that influence software quality can be classified into five, however as mention in previous section, this paper will only discuss two factors which are human and process.

Human

Agile implementation highly depends on human factor (Mnkandla, 2004; Lycett et al., 2003; Cockburn & Highsmith, 2001). Most literature highlight about team, project manager, developer and customer when discussing about human involved in Agile. Each of them needs to implement certain practices in sequence to ensure software quality. They are encouraged to be placed in a single space to facilitate intensive communication, because communication is emphasized for knowledge sharing (Pressman, 2005; Lycett et al., 2003; Highsmith & Cockburn, 2001). Besides, face to face communication is accepted as more effective to transfer idea compared to writing and reading documents (Highsmith & Cockburn, 2001). Smaller size team will encourage better communication, as more team members will cause the project to be 'less agile' (Abrahamsson et al., 2002). On top of that, most Agile methodologies does not work for large development teams. Thus, the suggested average number of team member is nine (Highsmith & Cockburn, 2001). The practices that must be followed are listed thoroughly in Table 1.

Process

Software development process is important in producing good quality software, as stated by Deming (1982) "*the quality of product is influenced by the quality of process used to develop it*". Besides the development process, project management also plays an important role. Basically Agile has all fundamental software development phases, which are requirement gathering, design, coding and testing. However, instead of having the four phases one after another throughout development, Agile has these phases iteratively in shorter time (Beck, 1999). Besides that, the implementation is also different. Requirement gathering and design in Agile is done iteratively and incrementally rather than gathering all requirements and designing upfront. During the development process, Agile emphasizes practices which can promote quality and faster delivery, such as pair programming and test driven development. These practices are further listed in Table 1.

Table 1. Agile Practices that Influence Software Quality

Factor	Sub Factor	Agile Practice	Reference
Human	Team	High competence and expertise	Tsun & Dac –Buu (2008)
		Great motivation	Tsun & Dac-Buu (2008)
		Common focus	Pressman (2005)
		Communication is used as important mechanism for knowledge sharing	Coram & Bohner (2005)
		Self-organized	Sliger & Broderick (2008)
		Co-located	Misra, Kumar & Kumar (2009)
		Empowered to make decisions	Sliger & Broderick (2008); Lindval et al. (2002)
		Decisions made collaboratively but speedy	Highsmith & Cockburn (2001)
		Able to give constant feedback	Highsmith & Cockburn (2001)
		Mutual trust and respect exist among team members	Pressman (2005); Highsmith & Cockburn (2001)
		Able to deal with ambiguity	Highsmith & Cockburn (2001)
		Intense interaction exist among team member	Highsmith & Cockburn (2001)
		Average size of team is 9 people	Highsmith & Cockburn (2001)
	Project Manager	Engaged with daily activities	Schuh (2005)
		Responsible to ensure that news (bad or good) is spread between customer and team.	Schuh (2005)
		Knowledgeable in agile process	Tsun & Dac-Buu(2008)
		Has adaptive management style	Tsun & Dac-Buu(2008)
		Responsible to the overall project's progress	Schuh (2005)
		Responsible to maintain relationship with customers	Schuh (2005)
		Acts more like a facilitator than a foreman	Sliger (2006); Schuh (2005)
		Responsible to build team cohesion	Sliger (2006)
	Developer	Able to respond quickly (responsive)	Cockburn & Highsmith (2001)
		Able to socialize (amicable)	Cockburn & Highsmith (2001)
		Able to work in group and spread knowledge	Cockburn & Highsmith (2001)
		Must be competent	Lindvall et al. (2002)
		Must be willing to learn continuously and work in changing situations	Schuh (2005)
		Must be inquisitive in nature	Schuh (2005)
	Customer	Able to give constant feedback	Lan & Ramesh, (2008);
		Able to communicate with the team	Rico et al. (2009)
		Able to present on-site throughout the development process(dedicated)	Paetsch et al. (2003); Highsmith & Cockburn (2001)
		Empowered to make decisions on behalf of other stakeholders	Boehm & Turner(2003); Paetsch et al. (2003)
		Know the business domain and knowledgeable	Schuh (2005); Boehm & Turner (2003)
		Do not feel afraid to be responsible to the decisions made	Schuh (2005)
		Willing to compromise	Schuh (2005)

Process	**Planning**	Done collaboratively with team members	Ambler (2010)
		Done continuously throughout the project at the beginning of each iterations and releases	Schuh (2005); Ambler (2005)
		Done according to features/ stories	Sliger & Broderick (2008); Schuh (2005)
		There exists daily stand up meetings among developers	Sliger & Broderick (2008)
		Release meeting is conducted at the beginning of project to create release plan	Sliger (2006); Schuh (2005)
		Iteration plan is created at the beginning of each iterations	Sliger (2006); Schuh (2005)
		User selects stories to be implemented in each iteration based on the estimates and velocity produced	Schuh (2005)
		Tasks estimation must be made by the developer who is going to accomplish the task	Wells (2009)
	Requirement Gathering	Iterative requirement engineering	Wells (2009); Lan & Ramesh, (2008)
		Face-to-face communication is emphasized instead of having written specification	Lan & Ramesh, (2008); Paetsch et al. (2003)
		High level requirements are written in user stories(XP)/ backlog(Scrum)/features(FDD and DSDM) for requirement gathering	Rico et al. (2009); Wells (2009); Lan & Ramesh, (2008)
		Detailed requirements are discussed in detail at each development cycle's start	Wells (2009); Lan & Ramesh, (2008)
		Information needed on user story: its name, the story and developer's estimation	Schuh (2005)
		The story written should be in simple and understandable language	Wells (2009); Schuh (2005)
		Prioritization to the user stories is done by user	Lan & Ramesh, (2008); Paetsch et al. (2003)
		Requirements can be reprioritized by user	Lan & Ramesh, (2008); Paetsch et al. (2003)
		Requirements can be added, removed or edited by users	Schuh (2005)
		Use prototype to validate the requirements	Lan & Ramesh, (2008)
	Design	Software is designed in small chunks and integrated in ongoing manner	Highsmith & Cockburn (2001)
		Agile Modeling is used to model high-level architecture of the system upfront	Ambler (2010)
		Unit tests which is implemented for Test Driven Development is used as detailed design artifact	Ambler (2005)
		Metaphor is used for determining architecture of the system	Rico et al. (2009)
	Coding	Implement collective code ownership	Ambler (2010); Rico et al. (2009); Wells (2009)
		Implement coding standards	Rico et al. (2009); Wells(2009)
		Implement pair programming	Rico et al. (2009); Wells (2009)
		Implement code and database refactoring	Ambler (2005)
		Unit tests are developed before the code is written	Ambler(2010); Wells (2009)
		Group and implement requirements with highest priority first	Leffingwell (2007)
		Ensure that the code produced is tested, working and integrated to system baseline	Wells (2009); Leffingwell (2007)
	Testing	Unit tests are developed before the code is implemented	Leffingwell (2007); Schuh (2005)
		Customer writes the user acceptance tests according to stories/features	Schuh (2005); Abrahamsson et al. (2002)
		User acceptance tests are used for requirements validation and verification	Lan & Ramesh, (2008); Paetsch et al. (2003);
		Continuous testing throughout development	Leffingwell (2007)
		Conduct review meetings to validate	Lan & Ramesh, (2008); Paetsch

		requirements	et al. (2003)
		Pair programming promotes peer review	Rico et al. (2009)

FUTURE WORK AND CONCLUSION

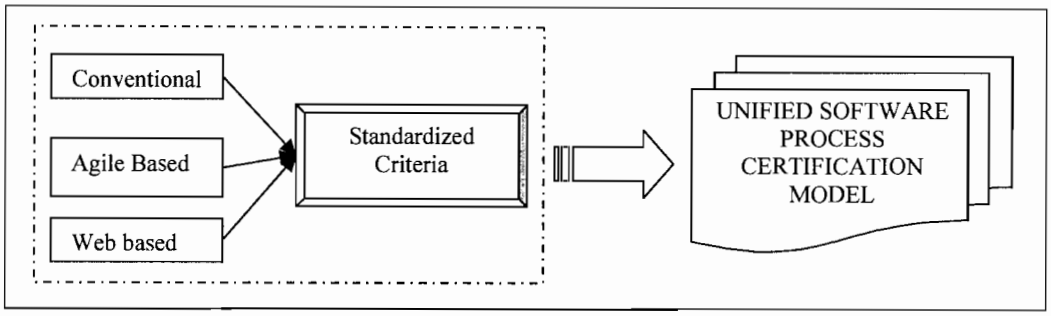

Figure 1. The Conceptual Framework for Unified Software Process Certification Model

Figure 1 shows the proposed conceptual framework of our research. As mentioned earlier, this study aims to construct a Unified Software Process Certification Model. One of the components in the model is the standardized criteria which act as a benchmark of the model. The standardized criteria will be constructed by identifying the best practices that must be performed in producing high quality software product. This study will identify and consider the best software development practices of conventional, Agile and Web-based software development. Since quality and rapid development process are treated as important goals to be achieved in software industry, Agile has been widely adopted by software developers. This paper presents the identified Agile software development practices that are related to human and process factors. Results of the literature found the practices emphasized for human factor are team work, effective communication, decision-making skills, and ability to work quickly. While for the process factor, the literature shows that Agile supports all fundamental software development phases although the implementation is different (refer Table 1). The importance of these practices will be verified by Agile practitioners through an empirical study. Only the practices which get high consideration from them will be included as the standardized criteria for our model. Moreover, during the empirical study, additional practices that might be suggested by the practitioners will be taken into consideration.

REFERENCES

Abrahamsson, P., Salo, O., Ronkainen, J., & Warsta, J. (2002). *Agile Software Development Methods Review and Analysis*. VTT Technical Report.

Agile Manifesto. (2001). Retrieved July, 7, 2010, from www.agilemanifesto.org

Ambler, S. (2010). Agile Project Planning Tips. Retrieved January, 7, 2011, from http://www.ambysoft.com/essays/agileProjectPlanning.html

Ambler, S. (2005). Quality in an Agile World. *Software Quality Professional,* 7(4), 34-40.

Arthur, L. J. (1993). *Improving Software Quality An Insider's Guide to TQM*. New York: Wiley Series.

Beck, K. (2000). *Extreme Programming Explained: Embrace Change*. San Francisco: Addison-Wesley.

Beck, K. (1999). *Embracing Change with Extreme Programming*, 70-77. doi: 10.1109/2.796139

Boehm, B., Turner, R. (2003). Observations on Balancing Discipline and Agility. *Proceedings of the Agile Development Conference*, 32-39. doi: 10.1109/ADC.2003.1231450

Cockburn, A., & Highsmith, J. (2001). *Agile Software Development: The People Factor*. 131-133. doi: 10.1109/2.963450

Conn, S. S. (2004). A New Teaching Paradigm In Information Systems Education: An Investigation And Report On The Origins, Significance And Efficiacy Of The Agile Development Movement. *Information System Education Journal* ,EDSIG.

Coram, M. & Bohner, S. (2005). The Impact of Agile Methods on Software Project Management. *12th IEEE International Conference and Workshops on the Engineering of Computer-Based Systems,* 363 - 370. doi: 10.1109/ECBS.2005.68

Deming, W. (1982). Out of Crisis. *Cambridge, MA: MIT Center for Advanced Engineering Study .*

Fauziah Baharom (2008). A Software Certification Model Based on Development Process Quality Assessment. Unpublished doctoral dissertation, Universiti Kebangsaan Malaysia.

Fuggetta, A. (2000). Software Process: A Roadmap, *Proceedings of the Conference on the Future of Software Engineering,* 25-34. doi: 10.1145/336512.336521

Hazzan, O., & Dubinsky, Y. (2009). Workshop on Human Aspects of Software Engineering. *Proceeding Of The 24th ACM SIGPLAN Conference Companion On Object Oriented Programming Systems Languages And Applications,* 725-726. doi: 10.1145/1639950.1639984

Highsmith, J., & Cockburn, A. (2001). *Agile Software Development: The Business Of Innovation.* 120-127.

Jamaiah Haji Yahya, Fauziah Baharom, Aziz Deraman & Abdul Razak Hamdan (2005). A Conceptual Framework for Software Certification. *KUTPM Journal of Technology & Management,* 99-111.

Krishnan, M. (1993). Cost, Quality And User Satisfaction Of Software Products: An Empirical Analysis . *Proceedings Of The 1993 Conference Of The Center For Advanced Studies On Collaborative Research.*

Lan, C., Ramesh, B. (2008). *Agile Requirements Engineering Practices: An Empirical Study.* 60-67.

Leffingwell, D. (2007). *Scaling Software Agility.* Boston: Addison-Wisley.

Lindvall, M., Basili, V., Boehm, B., Costa, P., Dangle, K., Shull, F. (2002). Empirical Findings in Agile Methods. *Proceedings of Extreme Programming and Agile Methods* , 197-207.

Livermore, J. A. (2007). Factors That Impact Implementing An Agile Software Development Methodology. *Proceedings of SoutheastCon 2007,* 82-86. doi: 10.1109/SECON.2007.342860

Lycett, M.,Macredie, R.D.,Patel, C., & J.Paul, R.(2003). *Migrating Methods To Standardized Development Practice.* 79-85. doi: 10.1109/MC.2003.1204379

Misra, S., Kumar, V., Kumar, U. (2009). Identifying Some Important Success Factors in Adopting Agile Software. *The Journal of Systems and Software,* 1869-1890. doi:10.1016/j.jss.2009.05.052

Mnkandla, E. (2004). *Balancing the Human and the Engineering Factors in Software Development.* 1207-1201. doi: 10.1109/AFRICON.2004.1406881

Nasution, M. F., & Weistroffer, H. R. (2009). Documentation in Systems Development: A Significant Criterion for Project Success. *Proceedings of the 42nd Hawaii International Conference on System Sciences,* 1-9.

O'Regan, G. (2002). *A Practical Approach to Software Quality.* Springer.

Paetsch, F., Eberlein, A. & Maurer, F. (2003). Requirements Engineering and Agile Software Development. *Proceedings of the IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises,* 308 - 313. doi: 10.1109/ENABL.2003.1231428

Pressman, R. S. (2005). *Software Engineering A Practitioner's Approach 6th Ed.* McGraw Hill.

Rico, D., Sayani, H., Sone, S. (2009). *The Business Value of Agile Software Methods.* Fort Lauderdale: J.Ross.

Schuh, P. (2005). *Integrating Agile Development In The Real World.* Hingham, Hingham: Charles River Media.

Sliger, M., Broderick, S. (2008). *The Software Project Manager's Bridge to Agility.* Boston: Addison-Wesley.

Sliger, M. (2006). A Project Manager's Survival Guide to Going Agile. Retrieved Novermber, 10, 2010, from http://www.rallydev.com/documents/rally_survival_guide.pdf

Tsun, C. Dac-Buu, C. (2008). A Survey Study Of Critical Success Factors In Agile Software Projects *The Journal of Systems and Software,* 961–971. doi:10.1016/j.jss.2007.08.020

Turk, D., France, R., & Rumpe, B. (2002). Limitations of Agile Software Process. *Proceedings of 3rd International Conference on Extreme Programming and Agile Processes in Software Engineering (XP 2002),* 43-46.

Verner, J., Liming, Z., Babar, M. A., & Ming, H. (2004). Software Quality and Agile Methods. *28th Annual International Computer Software and Applications Conference(COMPSAC'04),* 520-525.

Wells, D. (2009). Extreme Programming. Retrieved September 17, 2010 from http://www.extremeprogramming.org