The Risk Management in Software Development Using Agile Methods

Dr. M.V. Siva Prasad[1], Prof.V. Subrahmanya Sarma[2]
[1,2]*Professor, Anurag Engineering College, Kodad*

***Abstract*— The research about software projects failures shows that this industry has a lot to improve - especially concerning the Risk Management aspect of the projects. This study focus on the Agile Methods as a tool to help the Risk Management and the prevention of failures in software projects.**

***Keywords:* risk management, software development, agile methods.**

I. INTRODUCTION

The faults and failures in software projects compose an alarming statistic. One of the possible causes of this picture is the lack of risk management in the project process (CHAOS, 2013; MCMANUS, 2007), showing that neither managers nor methods are prepared to predict and/or solve a threat when it is materialized.
The same surveys show, especially in recent years, the growing use of Agile Methods as a project management tool (STATE OF AGILE, 2013; CHAOS, 2013). However, the question remains whether these agile adoptions will provide the managers tools for Risk Management that will help changing this scenario of recurrent failures.

Theoretical Framework

This section presents the definition of risk and its management in the software industry, followed by project failures statistics and how they are associated with the lack of risk control. Finally, Agile Methods and its main features are presented.

The Risk Management in Software Development

Risk may be defined as "a combination of the probability of a negative event and its consequences. If an event is inevitable but inconsequential, it does not represent a risk, because it has no impact. Alternatively, an improbable event with significant consequences may not be a high risk. These two factors are combined in what we experience as the possibility of loss, failure, danger" (GARG; BANSAL; SHARMA, 2014).
Relating this definition with software design situations, it may be said that risk is the possibility of not corresponding with the participant's expectations, in order to culminate in an unsatisfactory result. Examples are usually associated to non-compliances with the famous triad: scope, time and cost.
As software projects involve various classes of participants (customers, developers, users, stakeholders), each with different satisfaction criterias, it is known that the unsatisfactory results are multidimensional. For customers and developers, cost and schedule overruns are unsatisfactory. But for users, the unsatisfaction also relies on product features with errors, such as interface failures and performance or reliability deficits (BOEHM, 1991).
In this case, the software risk dimensions may be: (a) technical; (b) organizational; (c) environmental (BOEHM, 1991; MITTAL; BHASIN, 2013). The technical dimension results from the uncertainty in the design and implementation of tasks and procedures. The organizational dimension holds, for example, poor communication and inadequate organizational structure. The environmental dimension results from rapidly changing external environment and relationship issues with developers and/or users.
There are four ways to deal with project risks. The first is risk prevention that is achieved by reducing the project scope thus affecting product performance and competitiveness. Another way is to transfer the risk to a trusted third party but requiring guarantees (e.g. Service Level Agreements). It can also mitigate the risk by incorporating specific plans to deal with occurrences or minimize the damages. Finally is the risk acceptance, in which it is assumed sparingly and in a managed way (BOEHM, 2000; GHULE, 2014).

According to Higuera (1994), the following project management principles should be used to properly address the risk resolutions:

- Shared Product Vision: a shared vision of the product based on a common purpose, shared ownership, collective commitment and focus on results
- Teamwork: work cooperatively to achieve a common goal, sharing talents, skills and knowledge.

Global Perspective: context view on upper levels of the system, design and development. Recognize both the value of opportunities and the potential impact of adverse events, such as cost overruns, time delay or not fulfilling product specifications

- Prospective Vision: thinking about tomorrow, identifying uncertainties, anticipating potential results
- Open Communication: to encourage the free flow of information between all project levels, allowing formal, informal and improvised communication, using a consensus-based process that values individual voice (promoting shared knowledge and risk identification)
- Integrated Management: making risk management an integral vital part of project management, adapting its methods and tools for the project's infrastructure and culture
- Continuous Process: keep constant vigilance in order to identify and manage risks in all phases of the project life cycle.

Faults in Software Projects

For many years, the Standish Group has disseminated reports and researches worldwide on the performance of projects within the software industry. The appointed panorama is not good.

As seen in Table 1, from year 2004 to 2012, over 70% of software projects were delivered after the deadline, and around 50% were finished whith cost overruns.

Table 1 – Project Results in terms of time, scope and cost Source: CHAOS Report (2013)

Projects	2004	2006	2008	2010	2012
Time overruns	84%	72%	79%	71%	74%
Cost overruns	56%	47%	54%	46%	59%

In his research, John McManus (2007) reports a very close data to the one published by the Standish Group regarding the percentage of canceled projects or with cost and time above planned. The author also raised other statistics of cancellation causes of software projects that are seen in Table 2.

Table 2 – Reasons of Project Cancellation Source: John McManus and Trevor Wood-Harper (2007)

Business	Management	Technical
19,6%	53%	27,4%

It can be seen a consonance between the projects cancellation reasons raised by McManus and the risk dimensions mentioned by Boehm (1991) and Mittal (2013), in which the lack of Risk Management is presented as part of the Management group, holding most of the causes of failures (53%)..

Agile Methods

The Agile Methods appeared in the early 2000s as an alternative to the software development methods previously used, which were strongly plan oriented.

From its beginning, methodologies such as Extreme Programming (XP) and Scrum provided higher customer satisfaction, lower defect rates, reduced development time and the requirements adaptability (BOEHM; TURNER, 2003).

The common manual of principles to these methods is the Agile Manifesto (Beck et.al., 2001), which was a statement signed by 17 experts and that underlies the agile movement since then. Some values and practices recommended by the Manifest can be summarized in: communication, collaboration, pragmatism, adaptability, fast feedback loop, small iterations, velocity, simplicity, technical excellence, continuous improvement.

The Agile Methods are heavily based on the premise that the future is uncertain and that it is more valuable being ready to absorb the changes than to follow a preset plan. Another premise resulting from this, is the practice of experimentation and adaptation. Once the process is enlisted in small iterations (fast), and there is also a concern with the improvement, everything occurs within systematic experimentation and adaptation cycles.

METHODS

This paper is an exploratory research aimed to evaluate, using the deductive method (GIL, 1994), the presence of Risk Management a mid the use of Agile Methods in order to minimize the failure chances in a software project. The technical procedures were

literature research in publications such as articles, surveys, books, technical reports and web sites.

RESULTS AND DISCUSSION

Most of the fail projects indicated that the main threats could be avoided or greatly reduced if there was an explicit concern from the beginning in identifying and resolving the high risk factors (BOEHM, 1991).

On a cold analysis, none of the most popular Agile Methods and not even the Manifest itself directly addresses risk management in the system development stages.

However, as seen in Table 3, it can be affirmed that there is an inherent risk management in Agile Methods that is observed in the match between the principles raised by Higuera (1994) and the values of the Agile Manifesto.

Table 3 – Risk Management Principles and the corresponding Agile Manifesto Values

Risk Management Principles	Agile Manifesto Values
Shared Product Vision	Communication
Teamwork	Collaboration
Prospective Vision	Fast feedback cycle, Small Iterations
Open Communication	Communication
Continuous Process	Adaptability, Continuous Improvement

Another relevant factor that corroborates this statement is the Agile's central characteristic of experimentation and adaptation in daily practices. This makes the risk in actions and decisions naturally mitigated as we discover in advance the occurrence and the consequences of a threat, or the materialization of a loss but on smaller scale. This dynamic promotes an implicit risk management in Agile Methods.

CONCLUSION

Given what was presented, it concludes that the nature of Agile Methods naturally promotes risk mitigation actions, as seen in the likeness of risk management principles and the values of the Agile Manifesto. It is also known that neither agile or any other development method have all the answers, indeed containing a certain amount of risk (HIJAZI; KHDOUR; ALARABEYYAT, 2012). However, nothing prevents specific risk management techniques being employed together with Agile Methods (YLIMANNELA, 2011) for a better decision direction in relation to mitigation, prevention, transfer and risk acceptance in software projects, being subject for further work.

REFERENCE

[1] Beck, K. et. al. 2001. The Agile Manifesto. Available at http://agilemanifesto.org. (accessed date November 21, 2014).

[2] Boehm, B. 1989. Software risk management. 2nd European Software Engineering Conference University of Warwick, Springer, 387, 1-19.

[3] Boehm, B. 1991. Software Risk Management: Principles and Practices. IEEE Software, 32-41.

[4] Boehm, B. 2000. Project termination doesn't equal project failure. Computer, IEEE Computer Society 33: 94-96.

[5] Boehm, B., Turner, R. 2003. Using risk to balance agile and plan-driven methods. Computer, IEEE Computer Society 36: 57-66.

[6] Standish Group, 2013. CHAOS Report.

[7] Garg, R., Bansal, N., Sharma, P. 2014. Software Risk Management. International Journal of IT & Knowledge Management 11-13.

[8] Ghule S. 2014. Risk Analysis and Mitigation Plan in Software Development. International Journal of Engineering Science & Research Technology, 3(8).

[9] Gil, A. C. 1994. Métodos e técnicas de pesquisa social. 4 ed. Editora Atlas, São Paulo.