An Evaluation of Agile Software Methodology Techniques

A Sutharshan, S P Maj

Edith Cowan University, Perth, Western Australia

Summary
It is well documented that software projects are often over budget, over schedule and many fail to meet the functional requirements. In an attempt to address this problem numerous software methods have been introduced such as Extreme Programming (XP), Lean Development, Scrum etc. The main problem however has been to provide guidelines for efficient and effective team management. The Agile software philosophy was therefore developed. Uniquely Agile is a framework of principles that employs a range of different software methods. This approach allows the strengths of different software methods to be identified and aggregated. Hence a project manager can identify the best software method depending on the type of project.

Key words:
Agile philosophy, Agile methods, XP, Scrum, DSDM, FDD, Crystal, Lean, project management.

1. Introduction

Ever since firms started using computers to process their business data, successful implementation of information systems has been a concern of both researchers and practitioners [1]. In the early days of information systems, IT professionals alone were responsible for the information systems and staff in the rest of the organization took care of the business processes and their outcomes [2]. This arrangement was fine until businesses became more dependent on Information Technology for their operations. Meeting business needs became harder. There are numerous factors that can potentially handicap a successful IT department. Developing software systems is an expensive, and often a difficult process [3]. Although corporate expenditure on information technology (IT) has dropped in recent years, firms spend more than a trillion US dollars a year on IT [4].

Why are managing software projects so difficult? Why are we seeing so many project failures, especially in software development? Despite advances in software engineering, project failure remains a critical challenge for the software development community. Despite experiencing many successful projects, software engineers still struggle to ensure the consistent success of their projects. The history of failure of information systems development over the last 20 years is well recorded [5]. A survey of over 8000 projects undertaken in the year 2000 by 350 US companies revealed that one third of the projects were never completed and one half succeeded only partially, that is, with partial functionalities, major cost overruns, and significant delays [6]. Over the years, researchers have studied several aspects of software implementation, be it measuring success or developing and testing models that explain IS project success or failure. The need for the participation and involvement of users in IT development was recognized even in 70s [7]. Human related skills became important as a result of increased user involvement in the IS development process [8]. Cheney also identified the changing emphasis towards general interpersonal skills and, specifically, the ability to communicate with end users involved in the IS development process[8]. For a good software project to be successful, it has been indicated that focus should be placed on the processes, technology and people in order to achieve better performance, and the people-focus is by far the component that gets the least attention [9].
Software project management continues to be a challenging area for practitioners: more than half of all software projects experience severe difficulties and/or failure [10]. The Standish Group's "CHAOS Report," [10] a widely respected survey of software projects in industry and government, estimated that, in the year 2004, only 29% of software projects in large enterprises succeeded (i.e., produced acceptable results that were delivered close to on-time and on-budget). 53% were "challenged" (significantly over budget and schedule), and 18% failed to deliver any usable result. The projects that are in trouble have an average budget overrun of 56%. This represents a serious and chronic problem.

2. Agile Philosophy

Despite the existence of a wide range of different software methods organizations still find it difficult to deliver quality projects within time, budget and user expectations [11]. The main problems with project management have changed. During the 1980's the major factors were related to execution problems, during the 1990's the problem domain had significantly increased and included:

- lack of top management,
- commitment to the project,

Manuscript received December 5, 2010
Manuscript revised December 20, 2010

- failure to gain user commitment,
- misunderstanding the requirements,
- lack of adequate user involvement,
- failure to manage end user expectations,
- changing scope/objections,
- lack of required knowledge/skills in the project personnel,
- lack of frozen requirements,
- introduction of new technology,
- insufficient/inappropriate staffing,
- conflict between user departments

[12].

Significantly most issues identified are human related concerns. In an attempt to address this problem numerous software methods were introduced such as Extreme Programming (XP), Lean Programming and Scrum. Methods such as these were to some degree successful [13]. However no single method is available that can address all software project management expectations. For example XP encompasses pair programming but does not empower developers to make decisions.

Hence the Agile philosophy evolved based on four key values:

1. Individuals and interactions over processes and tools
2. Working software over comprehensive documentation
3. Customer collaboration over contract negotiation
4. Responding to change over following a plan

In recent years, processes based on the Agile Manifesto have been gaining acceptance among practitioners [14]. The principles behind this manifesto suggest that change should be welcomed at every stage of the software development cycle, that working software should be delivered frequently, and that conveying information via face-to-face conversation is more efficient than through written documentation [14]. Agile processes are characterized as informal and minimally documented, in addition, these processes put more emphasis on verbal and social communication on the development team [14].
Uniquely therefore Agile is a framework of principles that employs a range of different software methods – referred to as Agile methods. A 2003 global survey of experience using agile methodologies carried out by an Australian company produced the results that have been summarized below:

- 88% of organizations cited improved productivity
- 84% of organizations reported improved quality of software products
- 46% of respondents reported that development costs were unchanged using agile methodologies, while 49 percent stated that costs were reduced or significantly reduced
- 83% stated that business satisfaction was higher or significantly higher
- 48% cited that the most positive feature of agile methodologies was their ability to 'respond to change rather than follow a predefined plan' [13].

The benefits of agile are multi dimensional, but the most important change is that it focuses the entire organization on meaningful delivery to the customer. Agile methodology helps to achieve customer perceived value [15]. Agile software development methodologies have since their inception claimed to improve the quality of the software product [16].

Results from a survey done in 2006 at Microsoft to identify what the participants thought were the top 10 benefits with agile development are listed below in table 1 [17]. The top benefit was improved communication and coordination among team members. It was seen useful to bring testers and developers together. The second most cited benefit was quick releases. This was a consequence of continuous integration where workable software was released every few weeks than months or years.

No.	Benefits with agile development	Participant number
1.	Improved communications	121
2.	Quick releases	101
3.	Flexibility of design	86
4.	More reasonable process	65
5.	Increased quality	62
7.	Better customer focus	50
8.	Increased productivity	28
9.	Better morale	23
10.	Testing first	22

Table 1: Benefits to agile development methodologies [17]

The authors found that there is little comparative analysis of agile methods; some work have been done by Abrahamsson's group [18, 19]. However, to date it has not been possible to identify techniques unique or common to each Agile method.

3. Evaluation of Agile Methods

There are many Agile methods. For this study six of the most common Agile methods were selected:

1. XP
2. Scrum

3. DSDM
4. FDD
5. Crystal
6. Lean

Each of these methods was analyzed in detail and hence twenty five techniques were identified (table 2). The authors define a technique as a unique and important requirement in software development projects, for example iterative development, daily team meetings etc. Techniques were defined as general and specific based on how detail the techniques were defined in the methods.

	Technique	**General**	**Specific**
1	Daily builds of complete system		✓
2	Iterative development	✓	
3	Iteration of fixed length		✓
4	Incremental development	✓	
5	Customer on-site	✓	
6	Frequent delivery	✓	
7	Whole team works same location		✓
8	Dedicate meeting place		✓
9	Daily team meetings		✓
10	Testing is integrated	✓	
11	Project management emphasis	✓	
12	Communication	✓	
13	Collaboration	✓	
14	Coordination	✓	
15	Knowledge sharing	✓	
16	Working with uncertainty	✓	
17	Empowered to make decisions		✓
18	Courage to make mistakes		✓
19	Requirements as prototypes rather than text		✓
20	40 Hours week		✓
21	Pair programming		✓
22	Refactoring		✓
23	Small software product releases	✓	
24	Collective ownership of code	✓	
25	Champion role		✓

Table 2: List of techniques

Based on these techniques these six Agile methods were evaluated (table 3). Hence it is possible to identify the techniques unique to each Agile method. For example the technique specific to XP is '40 hours week' and to DSDM is 'dedicated meeting place'. There are other techniques which are common to limited methods. Scrum and FDD are characterized with technique 'champion role' and Scrum and DSDM are characterized with technique ' daily team meetings'. Techniques such as 'pair programming' will need to be investigated to identify suitability for the project and team.

A study was previously done comparing XP and Scrum using a framework based on the agile manifesto [20]. The study found to meet most, but not all of the criteria in the manifesto. When amalgamating two or more methods, it gives a solid basis. There are further practical reasons for combining methods. XP lacks support for project management [19], Scrum lacks specific practices for managing iterative and incremental projects. A combination of XP and Scrum [20], XP and Crystal methods [21], XP and ASD [22] are few of the proposed method combination that have been considered in the past. Only XP offers concrete guidance over whole lifecycle [19] and this explains why XP is the method most often proposed in combination with other agile methods. Recommendations to combine methods or use techniques from one method in another method have come from a need to address these weaknesses.

Hence a project manager can select a specific method or combination of methods best suited to the project.

Technique	***XP***	***Scrum***	***DSDM***	***FDD***	***Crystal***	***Lean***
Daily builds of complete system		✓				
Iterative development	✓		✓	✓	✓	✓
Iteration of fixed length			✓		✓	
Incremental development	✓		✓	✓	✓	✓
Customer on-site	✓					
Frequent delivery			✓	✓		✓
Whole team works same location	✓	✓			✓	✓
Dedicate meeting place		✓				
Daily team meetings		✓	✓			
Testing is integrated	✓		✓		✓	
PM emphasis		✓				
Communication	✓	✓	✓	✓	✓	✓
Collaboration	✓	✓	✓	✓	✓	✓
Coordination	✓	✓	✓	✓	✓	✓
Knowledge sharing	✓	✓		✓	✓	✓
Working with uncertainty	✓	✓				✓
Empowered to make decisions			✓			✓
Courage to make mistakes			✓			
Requirements as prototypes rather than text			✓			
40 Hours week	✓					
Pair programming	✓				✓	
Refactoring	✓					✓
Small software product releases	✓	✓	✓			
Collective ownership of code	✓				✓	
Champion role		✓		✓		

Table 3: Evaluation of Agile methods

Conclusions

The Agile methods have proved to be of value to the software development community. However there was no systematic method available to guide the selection of the most appropriate one or combination of for a given type of project. The identification of techniques allows each Agile method to be characterized and hence selected on the basis of the required techniques. This therefore is an aid to

improving project management. However further work is needed.

References

[1] Rivard, S., et al. Project Managers' Influence Tactics and Authority: A Comparison. 1998.

[2] Avital, M. and B. Vandenbosch. The relationship between psychological ownership and IT-driven value. in Proceedings of the twenty first international conference on Information systems 2000. Brisbane, Queensland, Australia

[3] Cerpa, N. and J.M. Verner, Why did your project fail? Communications of the ACM, 2009. 52(12): p. 130 - 134.

[4] Love, P.E.D., A. Ghoneim, and Z. Irani, Information technology evaluation: classifying indirect costs using the structured case method. Journal of Enterprise Information Management, 2004. 17(4): p. 312-325.

[5] Morien, R. Agile Management and the Toyota way for Software project management. in 3rd IEEE International conference on Industrial Informatics. 2005.

[6] Lamsweerde, A.v., Requirements engineering in the year 00: a research perspective, in Proceedings of the 22nd international conference on Software engineering. 2000, ACM: Limerick, Ireland.

[7] Lucas, H.C., A User-Oriented Approach to Systems Design, in Proceedings of the 1971 annual conference. 1971, ACM Press. p. 325 - 338.

[8] Cheney, P.H. Information Systems Skills Requirements: 1980 & 1988. 1988: ACM.

[9] Leonard, A. Enabling End Users To Be More Efficient During Systems Development. in Proceedings of SAICSIT 2002,. 2002: ACM.

[10] Standish, G., 2004 Third Quarter Research Report. 2004, The Standish Group International: West Yarmouth, MA, USA.

[11] Johnstone, D., S. Huff, and B. Hope. IT Projects: Conflict, Governance, and Systems Thinking. in Proceesings of the 39th Hawaii International Conference on System Sciences. 2006: IEEE.

[12] Keil, M., et al., A Framework for identifying software project risk. Communication of the ACM, 1998. 41(11): p. 76 - 83.

[13] Shine Technologies. Agile methodologies - Survey results. 2003 [cited 2010 27th May 2010]; Available from: http://www.shinetech.com/attachments/104_ShineTechAgileSurvey2003-01-17.pdf

[14] Valencia, R.E.G., V. Olivera, and S.E. Sim. Are Use Cases Beneficial for Developers Using Agile Requirements? in Fifth International Workshop on Comparative Evaluation in Requirements Engineering. 2007: IEEE.

[15] Gat, I. How BMC is Scaling Agile Development. in Proceedings of AGILE 2006 Conference. 2006: IEEE.

[16] Mnkandla, E. and B. Dwolatzky. Defining Agile Software Quality Assurance. in Proceedings of the International conference on Software AEngineering Advances. 2006: IEEE.

[17] Begel, A. and N. Nagappan. Usage and Perceptions of Agile Software Development in an Industrial Context: An Exploratory Study. in First International Symposium on Empirical Software Engineering and Management. 2007: IEEE.

[18] Abrahamsson, P., et al., Agile software development methods - Review and Analysis. 2002, University of Oulu.

[19] Abrahamsson, P., et al. New Directions on Agile Methods: A Comparative Analysis. 2003: IEEE.

[20] Visconti, M. and C.R. Cook, An ideal process model for agile methods, in 5th International conference on product focussed software process improvement PROFES. 2004, Springer-Verlag: Berlin. p. 431 - 441.

[21] Cockburn, A., Agile Software Development. 2002, Boston: Addison-Wesley.

[22] Highsmith, J., What is Agile Development? The journal of Defense Software Engineering, 2002.

Mrs. Anu Sutharshan is studying PhD at Edith Cowan University as a part time student. Her interest includes agile methodology, cultural studies, project management and team management. She has been in IT business for the past 20 years employed by Government and private sectors. She is currently working as Team Leader at Department of Transport, Western Australia, managing Application development and support team members. Anu has been involved in managing many complex medium to large software development projects.

A/Prof S. P. Maj has been highly successful in linking applied research with curriculum development. In 2000 he was nominated ECU University Research Leader of the Year award He was awarded an ECU Vice-Chancellor's Excellence in Teaching Award in 2002, and again in 2009. He received a National Carrick Citation in 2006 for "*the development of world class curriculum and the design and implementation of associated world-class network teaching laboratories*". He is the only Australian judge for the annual IEEE International Student Competition and was the first Australian reviewer for the American National Science Foundation (NSF) Courses, Curriculum and Laboratory Improvement (CCLI) program.