Title	A study of XP & Scrum: A Project Management Perspective
Author(s)	McHugh, Orla; Staunton, Christina; Slattery, Sarah; Rooney, Michael; Treacy, Frank
Publication Date	2008
Publication Information	Slattery, S., Rooney, M, Treacy, F., Staunton, C., McHugh, O. (2008) A study of XP & Scrum: A Project Management Perspective. In Proceedings of the 3rd International Business Informatics Challenge and Conference, Dublin City University, September 25th, 2008
Item record	http://hdl.handle.net/10379/1166

A STUDY OF XP & SCRUM: A PROJECT MANAGEMENT PERSPECTIVE

Frank Treacy, f.treacy1@nuigalway.ie [Student Author]

Michael Rooney, m.rooney2@nuigalway.ie [Student Author]

Sarah Slattery, s.slattery2@nuigalway.ie [Student Author]

Christiana Staunton, c.staunton1@nuigalway.ie [Student Author]

Orla McHugh, orla.mchugh@nuigalway.ie, 091 - 495283 [Supervisor]

Business Information Systems Group, Department of Accountancy & Finance,

National University of Ireland, Galway.

Abstract

This research examines agile software development from a project manager's perspective. It specifically investigates four different areas namely: project planning and scheduling; team selection; communication; and documentation on projects that use either Extreme Programming(XP) or Scrum as their software development methodology. Three organisations in Ireland participated in this study and one project manager from each organisation was interviewed. The size of the organisations, the level of experience of the project managers and the agile methodology in use varied across the organisation. The findings show that all project managers tended to be sceptical before implementing agile methodologies. However, post implementation they found that agile methodologies helped to improve their software development processes; provided the customer with a greater opportunity to work with the project team and also improved communication and collaboration amongst the project team. These elements contribute to a project team completing a more successful software development project.

1   INTRODUCTION

In today's world companies are demanding information systems that constantly meet their changing requirements (Duane et al., 1999, pg 13). The traditional plan-driven software development methodologies are not able to cope with these changing requirements and they lack the flexibility and fast pace development to meet customers needs (Sridhar et al., 2005). Statistically, traditional projects have failed as can be seen in Figure 1 below (The Standish Group, 1995). Factors influencing these project failures included lack of user involvement, proper planning, smaller project milestones, competent staff and a clear vision and objectives.
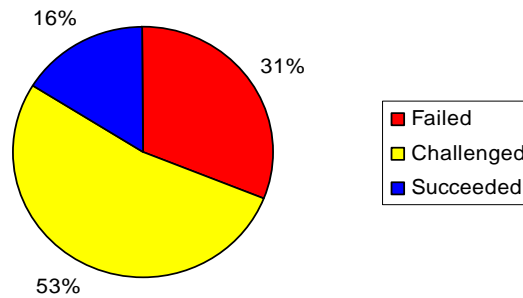
Figure 1 Chaos Report Findings (The Standish Group, 1995)

This has led to the development of an agile approach to software development, which also requires, project managers to adapt in order to successfully manage agile software development projects.

Project management of an agile software development project differs from that of a traditional software development project in several ways, which includes:

- Focusing on planning and scheduling using iterations as opposed to the traditional approach of planning and scheduling an entire project.
- Using teams that are smaller than traditional teams where team members are selected based on the skills required for the project.
- Including customers throughout the project and meeting on a daily basis for a short period of time in contrast to weekly meetings that are conducted on a traditional software development project (Cohn and Ford, 2003).
- Creating less but more useful documentation in comparison to traditional projects where large amounts of documentation can be created (Williams et al., 2003).

This project wishes to study the experiences of project managers that manage agile software development projects and how they handle project planning and scheduling; team selection; communication; and documentation which are the foundations of the Agile Manifesto as can be seen in section two below.

2   AGILE SOFTWARE DEVELOPMENT

Agile software development came about in 2001 when seventeen prominent figures from the software development industry came together in Utah, United States of America, to discuss ways of creating software in a lighter, faster, more people-centric way. The outcome of the meeting was the creation of the Agile Manifesto, widely regarded as the definition of agile development. The Agile Manifesto (2001) favours:

> "***Individuals and interactions*** *over processes and tools*
>
> ***Working software*** *over comprehensive documentation*
>
> ***Customer collaboration*** *over contract negotiation*
>
> ***Responding to change*** *over following a plan*"

While the Agile Manifesto acknowledges there is value in the items on the right, it places more emphasis on the items on the left" (Agile Manifesto, 2001).

The Agile Manifesto (2001) also set down twelve principles for the agile software development. These are to:

- Satisfy the customer by providing quality software on time.
- Allow changes to the project requirements at any stage of the project.
- Produce software more often by reducing software lifecycles.
- Have business people and developers working together on a daily basis for the duration of the project.
- Create an environment where people are motivated, supported and trusted.
- Have face to face communication throughout the project.
- Use working software as the primary measure of progress.
- Make sure all stakeholders are working at a similar pace so that everyone knows what each person is working on.
- Pay attention to detail leading to a quality product.
- Keep things simple and focus on the work that needs to be done.
- Have self-organised teams.
- Monitor the project team's progress and review how it can improve.

There are several different agile methodologies in existence for software development. Each agile methodology follows the principles defined above. However, the focus of this project is on two of the most popular agile methodologies, namely Scrum and Extreme Programming(XP) (Fitzgerald et al., 2006). A brief introduction to both of these agile methodologies is detailed below.

Extreme Programming (XP) was designed to deliver software that focuses on customer needs and customer satisfaction (Highsmith, 2002, pg 67 - 69). It is based on the premise that software projects should be dynamic, in that you should be able to adapt to changing requirements at any point during the project lifecycle. This means that it is a more realistic and a better approach to software development than attempting to define all the requirements at the beginning of a project as is done using traditional software methods. XP uses face to face communication in place of written documentation wherever possible and attempts to achieve customer satisfaction through iterations which are followed by user-testing and acceptance. There is sometimes written documentation for the software, yet according to Jefferies (2001, pg 1 - 8) code is rarely or never documented.

Scrum provides flexibility in the development process and the ability to respond to change where factors such as requirements, resources and technology can be unpredictable and complex resulting in the delivery of a useful system (Schwaber, 1995). It allows for changing requirements throughout the project which are completed using a series of short iterations or sprints. A Sprint may last from one week to one month and there may be three to eight Sprints in one systems development process before the system is ready for distribution. Schwaber and Beedle (2002, pg 36) suggest that a Scrum team should comprise of between five and nine people. A study at AG Communication Systems by Rising et al (2000) agrees with this and state a team should have no more than ten people. If more than ten people are available then multiple teams should be formed.

Schwaber and Beedle (2002, pg 2) and Fitzgerald et al (2006) state that XP and Scrum complement each other well, with XP providing support for technical aspects and Scrum providing support for project planning and tracking. Research by Fitzgerald et al. (2006) also shows that Scrum and XP can be tailored and customised to suit the needs of different projects as they learned from their research at Intel, Shannon.

2.1   Project Planning and Scheduling

In Scrum and XP, the underlying strategy of the selected agile methodologies is that of incremental development using a short development cycle (Leffingwell and Muirhead, 2004). This is evident in Scrum using the 30 day Sprint Cycle and in XP where a new release is made, often anywhere from daily to monthly.

Agile teams use three different levels of planning: release planning, iteration planning and daily planning (Cohn, 2006, pg. 28).

Release planning occurs at the start of a project to determine the scope, schedule and resources for each release. It is a very high level plan that covers the entire project. The project is broken into requirements known as user stories. Each user story is given story points or ideal days which reflect the size of the requirement. Ideal days is the estimated number of days to complete the user story whereas, story points are derived from a closed scale, for example, numbers 1 to 10, where 1 is the smallest user story and 10 is the largest user-story. For example, if it is known the team complete 40 story points for a fifteen day iteration, and the entire project is made up of 200 story points, it can be estimated that the project will require five iterations which is seventy five days.

An **iteration plan** is created in an iteration planning meeting. An iteration length is set and user stories from the release plan are prioritised to be developed within the iteration. Iterations can be timeboxed, which is the practise of fixing the iteration end date and not allowing it to change (Larman, 2004, pg 13). Iteration timeboxing is most commonly used in projects that require a large number of features that must be completed in a short space of time (Jalote et al., 2004). The amount of work which can be completed within each iteration is estimated based on the same approach as above, where the known or estimated velocity of the team per iteration is used. The velocity of the team per iteration is the number of days it takes them to complete one iteration. The team takes a closer look at each user story and details a list of tasks that will have to be undertaken to implement each user story. Estimates are carried out on each task, primarily by the person who is going to implement the task with the aid of relevant parties. Unlike release planning, estimates in iteration planning are based on time (in hours) as shown in Table 1 below. Iterations provide key stakeholders with visibility to the product throughout the development process, which encourages feedback and allows for more successful product development (Larman, 2004).

	Release Plan	Iteration Plan
Planning Horizon	3-9 months	1-4 weeks
Items in Plan	User Stories	Tasks
Estimated in	Story Points or ideal days	Ideal hours

Table 1   Primary Differences between a Release and an Iteration Plan.

Finally, there is **daily planning**. Most agile teams use some sort of daily stand-up meeting to coordinate work and synchronise daily efforts as can be seen in the daily scrum meeting. This daily planning is largely an informal process but teams make, assess, and review their plans during these meetings. In this meeting the team discuss what has been done since the last daily meeting and what will be done before the next daily meeting.

2.2   Team selection for a project

Project teams are made up of several roles, for example: stakeholders, business analysts, designers, a project manager, programmers and testers. Individuals within the team have a variety of differing personalities, skills, ability, knowledge and temperament. Therefore, it is important to choose people who are team players and who have good people skills, so they will work together to build on each others strengths and accomplishments (Cockburn, 2001, pg 60, pg 169).

The size of an agile project team can depend on the number of requirements and how large the project is. With larger teams of more than ten people it is harder for each member of the team to know what other members of the team are working on and this can sometimes result in the duplication of work. The main emphasis behind smaller teams is that they are easier to manage and easier to control. Hence, for larger teams there is a greater need for more efficient communication and co-ordination (Cockburn, 2001, pg 126 -136, Sanjiv et al., 2005).

During a project, requirements can change and new pieces of functionality can be added. This may result in the project becoming larger, which can require extra team members. Cockburn (2001, pg 126 - 127) states that as the size of an agile project increases, the cost of communication increases, face to face communication can be more difficult to arrange, the quality of communication can sometimes decrease and software may become more difficult to develop. Therefore, there is a major need for control of agile development and team selection. This is done in order to monitor peoples work so that work is not duplicated to ensure an agile project is under control (Cockburn, 2001, pg 126 - 127).

2.3   Communication

Communication is a dynamic process that individuals use to exchange ideas, experiences, and information through speaking, writing and body language. Good communication is at the core of successful projects. Open communication between team members and the project manager helps to make sure that project members are aware of potential risks and setbacks. Research shows that successful projects had project managers with good communication skills in contrast to failed projects which had project managers with poor communication skills (Ambler, 2002, pg 163 - 164, Sanjiv et al., 2005, The Standish Group, 1995). According to Ambler (2002, pg 159) communication is key to success on agile projects and is also one of the most crucial elements of both Extreme Programming (XP) and Scrum. In order for agile projects to succeed, the project team must create a highly communicative environment within which to work, as not doing this could increase the risk of failure (Ambler, 2002, pg 159 - 160).

Ambler (2002, pg 160 - 161) identifies several factors that affect communication within agile projects. These include:
- Physical proximity: the closer people are to one another then the greater the opportunities for communication.
- Temporal proximity: whether or not two people are working at the same time affects communication. Some people may be separated from co-workers by different time zones or by different working hours or schedules.
- Team Morale: the greater this is between the members of a team, then the greater the amount and quality of the information that is communicated.
- Tools: complicated software tools tend to act as barriers to communication.
- Anxiety about certain types of communication: Some people may prefer to speak on the phone or face to face, while others may avoid this type of communication and prefer e-mail. A method of communication that everyone is at ease with, and that will suit everyone for the duration of the project should be selected to overcome this anxiety.

Ambler (2002, pg 161 - 163) also identifies various different modes of communication for people to choose from when working together. These include: face to face at a white-board; face to face through having a conversation; video conversation; phone conversation; videotape; e-mail; audio-tape and paper. The most effective communication is face-to-face, especially through the use of white-boards as can be seen in Figure 2 below (Ambler, 2002, pg. 162).
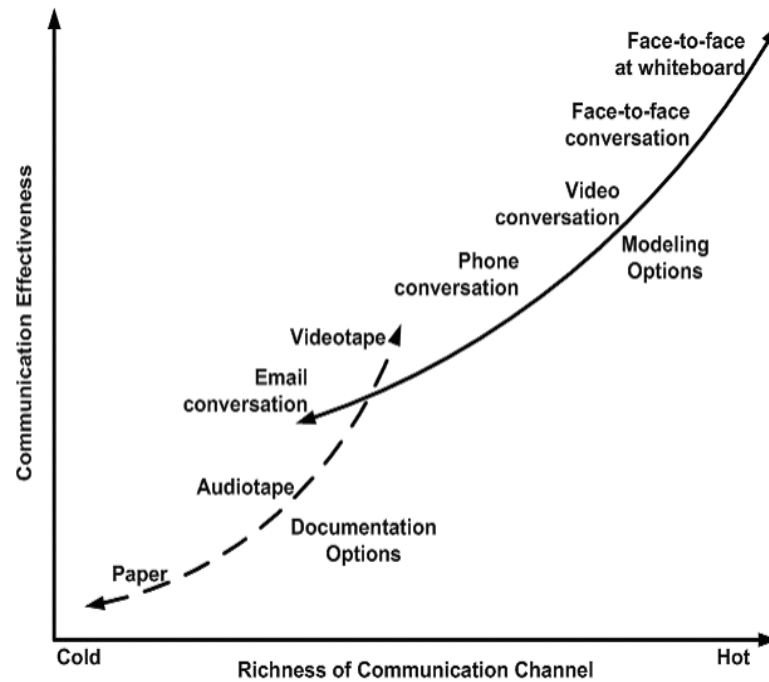
Figure 2   Communication Effectiveness (Ambler, 2002, pg. 162)

In summary, effective communication is an enabler of agile software development. It is critical that project teams are aware of all the different modes of communication available to them and they should pick the most suitable communication mode for their current situation. If a project is to succeed, the team must create a highly communicative environment and actively remove barriers to communication when they arise (Ambler, 2002, pg 164, Crushman, 1999, Sanjiv et al., 2005)

2.4   Documentation

Documentation can be seen as a form of communication between the various parties involved in a project. According to Ambler (2002, pg 241 - 242) a document is anything whose purpose is to convey information in a persistent manner. In a traditional project there is less communication so there is a much greater need for documentation. In agile projects the team relies on informal internal communication so the need for documentation decreases (Hass, 2007). Figure 3 displayed below compares the agile approach and the traditional approach to the generation of documentation (Ambler, 2002).
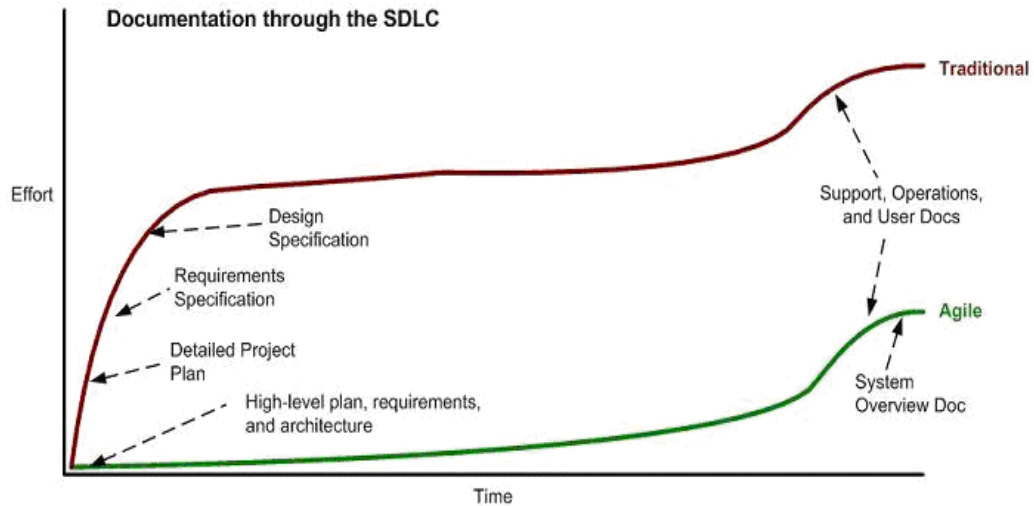
Figure 3   Difference between agile and traditional documentation (Ambler, 2002)

The agile approach is to delay the creation of all documents until as late as possible, creating them just before they are needed. By waiting to document information once it has stabilised it is possible to reduce both the cost and the risk associated with documentation. This does not mean that all documentation should be left towards the end (Ambler, 2002, pg 243). Notes should still be taken throughout development so that critical information is not lost or forgotten (Ambler, 2002, pg 244, Cockburn, 2001). There are also other reasons as to why documentation should be created for agile projects. These include:

- To support communication with an external group: it is not always possible to co-locate a development team or to have project stakeholders available at all times.
- To think something through: writing ideas down on paper can help to solidify them and helps people identify where potential problems may occur.

Agile developers recognise that it can be difficult to have just enough documentation, at the right time, for the right audience. A project team should concentrate on creating documentation that provides maximum value to its customers. An agile document just needs to be good enough for its intended audience. Although agile software development produces working software over comprehensive documentation that does not mean that documentation should not be created nor neglected (Ambler, 2002, pg 254, Cockburn, 2001).

3   RESEARCH APPROACH

Case study was the research strategy adopted for this project. Multiple case studies were conducted in order to obtain a broader view of practices in several organisations and also to allow for comparison and analysis across different organisations. Additional cases studies would have been conducted if more time was available for the project. Other research approaches such as a survey method may have been used, but due to the limited timescales for this research project it may have been difficult to obtain access to the number of organisations required for quantitative analysis.

3.1   Selection of Organisations

The main criteria used for the selection of organisations was to identify project managers working in organisations based in Galway (due to their proximity to the university) who were currently managing an agile project using either XP or Scrum. The team contacted a large number of companies based in

Galway to determine if they were using agile methodologies. Three project managers were identified that were willing to participate in the research. Each of these project managers were working on teams that used XP or Scrum as their methodology for developing software.

3.2   Data Collection

As the project team wished to gather detailed opinions and perspectives from each project manager semi-structured interviews were used to collect data. A number of open-ended and closed questions relating to each section detailed above were developed. The open-ended questions gave project managers the opportunity to state their opinion. The team ensured that the questions asked were not leading or biased. On completion of all the interviews the findings from each company were analysed and compared. These findings are presented in the next section.

4   FINDINGS AND ANALYSIS

This section contains the findings and analysis of the case studies. Research interviews were conducted in February 2008.

4.1   Company Background

Three organisations participated in the research and one project manager from each organisation was interviewed. One organisation did not wish to be named in this research, so the organisations are listed as Company A, B and C. Details of the participating organisations and the project managers interviewed are displayed in Table 2 below. Views are representative of the opinion of the project manager interviewed only.

	Company A	Company B	Company C
Industry of Organisation	Information Technology	Information Technology	Financial Services
No. of employees on site	35 (on site)	600(Ireland)	185 (on site)
Years of experience as Project Manager	2	11	6
Years of experience managing Agile Projects	2	5	10 months
No. of Agile Projects Managed	2	3 with numerous releases.	4
Customers	External	Internal	Internal
Co located project teams	Yes	No	No
Agile Methodologies Used	XP (Customised)	XP (Customised)	Scrum & XP (Both Customised)

Table 2   Company and Project Manager Information

The project manager interviewed in Company A manages a team consisting of four developers, and one tester. The team is made up of three senior and two junior people. In Company B the project manager currently manages a team of eight members, which consists of subject matter experts and individuals who represent business operations and IT personnel. Company C has a team director and

technical architect based in the United States. The current project team consists of three team leads, two junior developers, a user interface designer, a business analyst, and an individual who looks after Quality Assurance.

4.2 Project Planning and Scheduling

All three companies used the practise of time boxing in their projects. Company A, B and C set the duration of iterations as seen in Table 3 below.

	Company A	Company B	Company C
Iteration Length in Days	5-10	10	15

Table 3   Iteration Length of each company.

It was found that Company A vary their iteration size between 5-10 days depending on the project and how much work is involved compared to Companies B and C who use the same iteration length for all projects. The project manager in Company B stated that 5 days is to short because not enough product is achieved in that time period and felt that 15 days was to long. The project manager in Company C initially started with using 10 day iterations but struggled to produce meaningful functionality in that time period so the iterations were changed to 15 days.

All three companies used estimation before they start working on an iteration. In Company A their external customers discuss the requirements with the team and estimates are made using comparisons of previous pieces of work. In Company B and C, where the customers are internal, estimation is a collaborative effort amongst the team.

Company C also estimates the size of a user story. This is done by means of a scale where a user story is given a certain number of points based on its size and work involved. For each iteration the points of all the user stories within an iteration are summed. Then the total points are compared to the natural velocity of Company C's 15 day iteration. For example, in Table 4 below, iteration X contains seven distinct user stories and the total point for the iteration is 65. The project manager knows from previous iterations that their natural velocity of iteration is 70 points so they are comfortable that iteration X will be completed in their 15 day iteration.

Iteration X	Points
User Story 1	5
User Story 2	10
User Story 3	6
User Story 4	7
User Story 5	15
User Story 6	13
User Story 7	9
Total	65

Table 4   User Story Points

The project managers in all three companies felt that agile estimation was more accurate than estimation methods used during a traditional SDLC. Agile estimation was tracked on a day-to-day basis where estimates are continuously assessed and improved by the project manager. In the case of

Company C, the project manager meets with each developer on a daily basis and as a result this close, hands-on tracking of estimates resulted in increased control of the project as opposed to the traditional SDLC where estimates are given at the start of the project and may not be accessed until later stages of the project.

In all three companies requirements were prioritised by the customer. In Company A, where the customer is external, prioritisation is solely the responsibility of the customer. In companies B and C, where the customer is internal, the customer drives the prioritisation while input and feedback from the project team is taken on board and often results in changes in the prioritisation. It was noted by Company C, that it is the ideal for the customer to prioritise requirements as it leads to a more usable product, but customer involvement is not always possible where the project is technology driven.

4.3   Team make-up

In all three companies, teams are chosen depending on the skill set required and who is available at the time. For all three companies the size of the team required depends on the project, the amount of functionality it requires, and the time and effort needed.

Company A finds it quite hard to divide up tasks as they lack experience, but the team use user stories as much as possible to help them. Company B and C have experts in each area working on the tasks. In company C, where the developers are very experienced in their areas, tasks are divided on the basis of expertise in a certain area and the manager monitors the tasks each member of the team is working on.

None of the companies interviewed experienced any resistance from team members when using agile methodologies.

4.4   Communication

The project manager in Company A stated that the company manages communication by having weekly team meetings where the progress of development is discussed. The project manager in Company B stated that communication on agile projects varies from project to project and it is managed differently depending on whether the team is onsite for the project or distributed across different countries. Company C uses a variety of communication tools and technologies including instant messenger, audio and video conferencing. All three companies agreed that communication is much easier to manage when the team and the customer are co-located and acknowledge that face to face communication is the most effective means of communication when working on an agile project.

Company A finds face to face the most efficient way to communicate as there is better interaction between the team itself and between the team and the customer. In Company A, the project team is co-located so communication mostly is face to face along with email, and instant messenger.

Company B uses face to face communication where possible along with emails, documents, and conference calls. On their current project in Company B, the project team is spanning three countries in Europe and this poses a challenge for the team in how communication is handled amongst the team.

Company C's project team is not co-located. As team members are based in Ireland and the United States, daily stand up meetings take place at 4.45pm in Ireland so that it is a suitable time for team members in the United States. Company C finds that when communicating with team members in the States, it is best to use telephone conversations or video and audio conferencing to communicate with team members abroad and follow up these conversations with a summary of the conversation in an email for future reference.

Factors which affected communication were mainly due to different time zones and whether or not the project team and customer were co located. All three companies acknowledged that it is easier to manage an agile project when the team is co-located as communication can be face to face and instant.

Another issue which affected communication is culture differences and language barriers. Company B stated that not everyone's native language is English and that some times team members from other countries can have difficulties understanding what another team member has said or written.

The main differences in communication between an agile project and a traditional project are as follows: Company A stated that communication takes place on a more regular basis and it involves the customer more than on a traditional project; Company B found that communication was easier on an agile project and that communication is more verbal than written; and Company C also found that communication happens on a daily basis rather than every three to four days as it would have happened on a traditional project.

4.5   Documentation

Company A manage their documentation through the use of a Sharepoint site and also through designing a project plan which contains user stories, releases and iterations. Each developer writes up the documentation which corresponds with the piece of functionality they are working on.

Company B had to meet IS9000 requirements, such as a development plan, test plan and design specification. In this company documentation was seen as a task to be completed rather than something which could be useful and the focus was on getting the documentation done and signed off rather than producing useful documentation.

The manager in Company C stated that his team struggled when it came to writing up documentation, so it was decided to write up the documentation at the end of a project. However, sometimes high level documentation is completed at the start, but generally the majority of the documentation, from a technical perspective, is done at the end. They feel that it is better to build something and then document it rather than document something and then build it, as requirements change and so do parts of functionality; which can result in documentation becoming out of date. The manager in Company C allows the people working on certain functionality to produce the documentation when their piece of functionality is finished and then the manager signs off on the documentation before it is sent off to the customer to be reviewed. Company C struggled with their documentation as they did not see it as a high priority.

All three companies strongly agreed that there was a much larger volume of documentation involved when working in a traditional project in comparison to an agile project.

4.6   Post Implementation

All three companies were sceptical of agile methodologies from the outset. However, these perceptions changed as time went on and the people within the agile teams became more familiar with agile methodologies. In all three cases it is clear that the project manager preferred using an agile methodology to the traditional waterfall approach.

5   DISCUSSION AND CONCLUSION

The purpose of this section is to discuss the findings and consider the conclusion from the research conducted. This section will conclude by considering the implications for practice.

5.1   Discussion of Findings

On completing this research project, the project team found that there were similarities and differences between the existing literature and what was obtained from the project managers interviewed in relation to agile methodologies and project management.

After implementing agile methodologies, the project managers and the agile project team found that it improved communication and collaboration amongst project team members and documentation was light yet relevant.

Fitzgerald et al (2006) argue that Scrum and XP can be tailored and customised to suit different projects. The findings of this research concur with this as all three project managers were using customised versions of the methodologies to fit in with their organisational needs and management issues.

Larman (2004) states that iterations provide key stakeholders with greater visibility to the product throughout the development process, which encourages feedback and allows for more successful product development. This research found that visibility is not always apparent where projects are largely technology driven.

In the literature it stated that teams needed to be co-located (Ambler, 2002, pg 161 - 162). However, two of the three project managers interviewed had teams working across different regions. While the project managers admitted that it was not ideal for the project team to be dispersed, it still worked for them. Ambler (2002, pg 160 - 161) identifies that physical proximity can affect communication as the closer people are to one another then the greater the opportunities for effective communication. The findings concur with this as all project managers agreed that communication is much easier to manage when the team and customer are co-located and acknowledge that face to face communication is the most effective means of communication when working on an agile project.

Ambler (2002, pg 243) argues that all documentation should not be left towards the end. However, the findings show that documentation was seen as a task rather than something which could be useful. The focus was on getting the documentation done and signed off rather than producing useful documentation. As a result teams can struggle when it comes to writing up documentation so it was decided to write up the documentation at the end of a project.

5.2   Implications for Practice

It is hoped that this research will provide readers with an insight into managing software projects that use XP and Scrum from a project management perspective. This research obtains the opinions of project managers in relation to two agile methodologies and identifies what agile practices project managers are currently using in their organisations. As a result the findings in this research are limited somewhat, as it can only be applied to these two agile methodologies. Other organisations that use these agile methodologies may have different experiences to those related in this document. The four areas that were under investigation in the research were: project planning and scheduling; team selection; communication; and documentation. The research also identifies issues that organisations had when using an agile methodology. These issues may help other organisations to decide which agile methodology is most suitable to their needs.

References

Agile Alliance (2001)."Agile Alliance". Retrieved 14/12/2007, from www.agilealliance.org

Agile Manifesto (2001)."Agile Manifesto". Retrieved 13/12/2007, from www.agilemanifesto.org/history.html

Ambler, S. (2002) *Agile Modeling: Effective Practices for Extreme Programming and the Unified Process,* Wiley.

Cockburn, A. (2001) "*Agile Software Development ",* Cockburn & Highsmith Series Editors.

Cockburn, A. and Highsmith, J. (2001) "Agile Software Development: the business of innovation" *IEEE Computer,* **Vol 34**(9), pg 120.

Cohn., M. (2006) *Agile Estimating and Planning,* Prentice Hall.

Crushman, R. (1999) *The Standish Group -Chaos: A Recipe for Success..*

Duane, P. T., Richard, B. and Heinz, K. (1999) *"Growing systems in emergent organisations" Commun. ACM,* **Vol 42** (8)**,** pg117-123.

Fitzgerald, B., Hartnett, G. and Conboy, K. (2006) *"Customising agile methods to software practices at Intel, Shannon", European Journal of Information Systems,* **Vol 15**, pg 200–213.

Hass, K. (2007) "*The Blending of Traditional and Agile Project Management", PM World Today,* **Vol 9** (5).

Highsmith, J. (2002) *Agile Software Development Ecosystems,* Addison Wesley.

Jalote, P., Palit, A., Kurien, P. and Peethamber, V. T. (2004), "Timeboxing: a process model for iterative software development", *Journal of Systems and Software,* **Vol 70** (1)**,** pg 117-127.

Jefferies, R. A., Ann; Hendrickson, Chet (2001) *Extreme Programming Installed,* Prentice Hall PTR, , Upper Saddle River, NJ.

Larman, C. (2004) "*Agile and Iterative Development, A managers guide",* Addison-Wesley.

Leffingwell, D. and Muirhead, D. (2004) "*Tactical Management of Agile Development: Achieving Competitive Advantage*" In *Rally Software Development Corporation.*

Rising, L., Rising, L. and Janoff, N. S. (2000) "*The Scrum software devlopment process for small teams*" *Software, IEEE,* **Vol 17**(4), pg26-32.

Sanjiv, A., Bob, P., Fred, S. and Susan, W. (2005) "Agile project management:steering from the edges" *Commun. ACM,* **Vol 48** (12), pg 85-89.

Schwaber, K. (1995) "*Scrum Development Process*" In *Workshop on Business Object Design* OOPSLA'95.

Schwaber, K. and Beedle, M. (2002) *Agile Software Development with Scrum,* Prentice Hall.

Sridhar, N., RadhaKanta, M. and George, M. (2005) "Challenges of migrating to agile methodolgies" *Commun. ACM,* **Vol 48** (5), pg 72-78.

The Standish Group (1995) "*The Standish Group - Chaos*".

Williams, L., Williams, L. and Cockburn, A. (2003) "*Agile software development: it's about feedback and change*" *IEEE Computer,* Vol 36(6), pg 39-43.