Spring 3-23-2010

AGILE DEVELOPMENT – SCRUM ADOPTED IN PRACTICE BUT NOT IN PRINCIPLE

Kiriaki Flouri
University of Wales Institute Cardiff, kflouri@uwic.ac.uk

Hilary Berger
University of Wales Institute Cardiff, Hberger@uwic.ac.uk

AGILE DEVELOPMENT – SCRUM ADOPTED IN PRACTICE BUT NOT IN PRINCIPLE

Kiriaki Flouri
Cardiff School of Management, University of Wales Institute Cardiff, Colchester Avenue, Cardiff, CF23 9XR, UK
Email: kflouri@uwic.ac.uk

Dr. Hilary Berger
Cardiff School of Management, University of Wales Institute Cardiff, Colchester Avenue, Cardiff, CF23 9XR, UK
Email: Hberger@uwic.ac.uk

Abstract

The move to agile software development methodologies has generated great enthusiasm. The emphasis on team-oriented development and reliance on people rather than predefined processes is transforming software development into a socio-technical process. Through the lens of a real-world project we examined the difficulties experienced when an IS development project shifted from a structured waterfall approach for upfront requirements gathering to a Scrum agile approach for the development activities. We specifically look at the agile values and principles of 'people, working software, end-user involvement and responding to change'. Although the transition was successful in practice, in principle the project failed. The empirical case study evidences the characteristics involved and we put forward critical factors of the preparation of the environment (i.e. adequate Scrum training), effective communications (i.e. consensus on a standard working context and sufficient time for testing), optimal team structure (i.e. personalities) and effective team leadership to inform future development practice.

Keywords: Agile Methods, Scrum, People, Working Software, End-users, Change.

1.0 Introduction

The shift from former traditional structured development approaches such as the Waterfall Model to agile software development methodologies is well documented. However whilst the move from the former static traditions to the more dynamic nature of agile development practices improved the discipline of systems development it did not resolve all the problems associated with the need to accommodate business uncertainty and the changing requirements of stakeholders/users (Elliott 1997; Graham 1989). Continued high rates of project overrun, over budget failures and systems that did not meet user requirements remained challenging (Boehm 1999; Coughlan and Macredie 2002; McConnell 1996). Consequently growth and change are recognized as intrinsic elements of IS development thus creating a demand for alternative more flexible development approaches able to respond to the increasingly dynamic nature of evolving business environments – namely agile approaches (Elliott 1997; Martin 1991; Raffoni 2000).

A number of agile software development methodologies or lightweight methodologies have been developed since 1990s to embrace, rather than reject, high rates of change (Abrahamsson *et al.* 2003; Boehm 2002; Williams and Cockburn 2003). Some examples are Adaptive Software Development (ASD, Cockburn 2000), Dynamic Systems Development Method (DSDM, Stapleton 1997), eXtreme Programming (XP, Erickson *et al.* 2005, Lindstrom and Jeffries 2004), Feature Driven Development (FDD, Palmer and Felsing 2002), Rational Unified Process (RUP, Kruchten 2000), and Scrum (Schwaber and Beedle 2002).

Agile software development methodologies are characterized by short iterative and incremental development cycles, time-boxed development, user involvement, collaborative decision-making, incorporation of rapid feedback and change, frequent delivery into the system under development (Cockburn and Highsmith 2001; Highsmith 2002; Stazinger *et al.* 2005). The focus is on prioritizing the core system functionality that is required and then delivering additional functionality in later iterations (Lindvall *et al.* 2002). Team-oriented development is emphasized where there is a reliance on people rather than predefined processes similar to a socio-technical process that optimizes both technical and social aspects where the goal is to enable future users to play a major part in the design of the system (Iivari *et al.* 2000).

Dyba and Dingsøyr's (2008) systematic review of empirical studies of agile software development emphasizes that there is a clear need to increase both the number and the quality of studies on agile software development. In particular, agile project management methods, such as Scrum warrant further attention which they believe is the most under-researched.

Scrum is one of the more widely used agile methods and its first references in the literature point to the article of Takeuchi and Nonaka's work (1986) for managing the systems development process. *Scrum is lightweight agile project management method based on small, empowered, self-organizing teams; complete visibility; and rapid* *adaptation* (Leffingwell 2007: 41). It has found success with numerous organisations such as British Telecom and Siemens (Sutherland *et al.* 2007). For example, it is used on some of the world's largest projects at British Telecom and Siemens because of its high productivity with distributed development teams. It is the only software development process that has repeatedly demonstrated linearly scalable productivity when adding resources to large projects (Sutherland *et al.* 2007*).* Although Scrum does not prescribe any specific software development techniques for the implementation phase it concentrates on team collaboration, timely and empowered decision-making and the accommodation of business changes to provide development flexibility in a dynamic, unpredictable and complex environments (Schwaber and Beedle 2002).

Through the lens of a recent real-world project involving an International Educational Foundation we examine how a development project used a structured waterfall approach for upfront requirements gathering and then adopted the values and principles of agility for the development activities utilizing a Scrum approach. It is not the remit of this paper to address the rationale of the preceding traditional waterfall approach or document the transition but rather to investigate the difficulties experienced that prevented success of the case study.

We use the ‘Universities and Transcripts Project’ as a case study to examine the problems experienced with the agile approach in terms of the values and principles documented by the Agile Alliance Manifesto i.e. ‘people, working software, end-user involvement and responding to change’ (Beck *et al.* 2001). Although successful in principle the empirical case study evidences the difficulties that the stakeholders experienced accommodating the transition process and in participating in the agile activities ultimately leading to the failure of the project.

This paper is organized as follows, in the next section we set out the theoretical context of the research case study, we then present the research philosophy and methods, describe the case study context, put forward the data analysis and research findings and finally present our conclusions and set of critical factors.

2.0 Theoretical Context

In this section we look at the core values and principles of agile development practices and examine the Scrum agile approach adopted.

2.1 Agile Development - Core Values and Principles

As previously mentioned the move towards agile development practices evolved to accommodate problems associated with former structured development approaches (Elliott 1997; Graham 1989). The aim was to address the continued high rates of project overrun, over budget failures and systems that did not meet user requirements (Boehm 1999; Coughlan and Macredie 2002). Agile methods emerged to respond to the demand for alternative more flexible development approaches able to cope with the increasingly dynamic nature of evolving business environments (Elliott 1997; Martin 1991; Raffoni 2000).

In 2001, 17 prominent figures in the field of agile development (or 'light-weight methods') came together to discuss ways of creating software in a lighter, faster, more people-centric way. They coined the term 'agile software development' and formed the Agile Alliance establishing a manifesto of principles. This manifesto is widely regarded as the canonical definition of agile development and its principles (Dyba and Dingsøyr 2008; Cockburn 2002; Highsmith 2002; McAvoy 2007). The manifesto states that agile development should focus on four core values:

• individuals and interactions	vs.	processes and tools
• working software	vs.	comprehensive documentation
• customer collaboration	vs.	contract negotiation
• responding to change	vs.	following a plan

The items on the left hand side are valued more highly that those on the right side. These represent the values and principles of 'people, working software, end-user involvement and responding to change' that the case study is concerned with (Beck *et al.* 2001). These core values are underpinned by a number of accompanying principles, for example - self-organizing and motivated teams; the frequently delivery of useful software; face-to-face communication, accommodation and regular adaptation of business changes; co-operative collaboration between business people (i.e. sponsors, end-users and so on) and developers. They are typically described as a cohesive collection of practices.

When taken in turn there is literature to support these statements. Firstly, people are regarded to be of greater value than tools and processes. A tenet of agility is the active engagement of stakeholders throughout the project. Such involvement increases understanding and commitment to the project, increases acceptance, reduced training needs and results in the right system being developed (Barki and Hartwick 1994; Carnall 2003; McConnell 1996). People have been recognized as the primary drivers of project success in agile methods because a good and rigorous process and the right tools will not save the project if the development team does not have skilful members (Cockburn 2002; Nerur *et al*. 2005; Martin 2003; Vinekar *et al.* 2006).

Secondly, McMahon (2005:1) advocates the value of working software that is frequently demonstrated to customers via short development iterations. It is believed that '*a better way to ensure customer needs are met is through working software rather than through formal written words*'. In other words, working software without documentation is better than non-working software with volumes of documentation. In agile methods where working software is delivered early and often then documentation can be added at a later date. However Cockburn (2002) questions what is the right amount of documentation? He believes that the right amount can be described as 'just enough' and 'barely sufficient' also advocating working software.

Thirdly, value statement concerns customer collaboration over contract negotiation. This is a fundamental principle across all the agile approaches. With Scrum it is achieved through interaction between developers and users to articulate how they (end-users) use the IS system – developing User Stories (Cohn 2004). Thus by working closely with the end-users regular feedback can be incorporated that will contribute to a successful project development.

Finally, responding to change over following a plan. Since, change is inevitable in software development (Pressman 2001) developers must be able to react to change when it happens. Plans are important, but the problem is that software projects cannot be accurately predicted far into the future due to the many variables involved (Koskela 2003). These four values provide the nimbleness needed to survive in a turbulent business world. The idea is to find the right balance between these items.

To sum up, the implication is that formalization of the software process hinders the human and practical component of software development, and thus reduces the chance for success. While this is true when formalization is misused and misunderstood, one has to be very careful not to overemphasize and under-measure the items on the left hand side since this can lead to the same problem, poor quality software. The key is finding the right balance (Boehm and Turner 2003).

2.2 Scrum Methodology

The term Scrum originally derives from a strategy in the game of rugby, in which fifteen players on two teams compete against each other, and it denotes 'getting an out-of-play ball into the game' with teamwork (Schwaber and Beedle 2002). Takeuchi and Nonaka (1986) first used rugby strategies to describe hyper productive development processes. Three strategies from rugby including a holistic team approach, constant interaction among team members, and unchanging core team members are adopted into Scrum management and control processes. The focus of Scrum is project leadership and requirements management (Schwaber and Beedle 2002) in situations where it is difficult to plan ahead. Scrum defines a high-level life cycle for construction iterations where software is developed by a self-organizing team in increments (see Figure 1 below).

Scrum involves daily stand-up Scrum meetings and short, time-boxed iterations called Sprints. An appointed Product Owner decides which backlog items should be developed in the following sprint employing a user prioritized requirements list (the Product Backlog of the features to be implemented). Empowered team members coordinate their work in the daily stand-up meetings. One team member, the Scrum Master, is in charge of solving problems that prevent the team from working effectively to achieve the frequent delivery of working software (Dyba and Dingsøyr 2008; Schwaber and Beedle 2002)

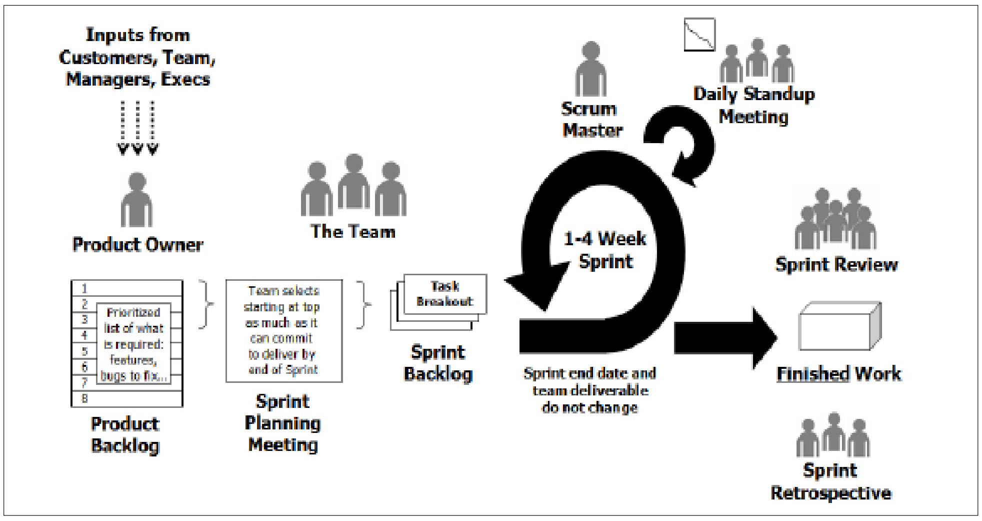
Inputs from Customers, Team, Managers, Execs
Product Owner
The Team
Scrum Master
Daily Standup Meeting
1-4 Week Sprint
Sprint Review
Prioritized list of what is required: features, bugs to fix...
Product Backlog
Team selects starting at top as much as it can commit to deliver by end of Sprint
Sprint Planning Meeting
Task Breakout
Sprint Backlog
Sprint end date and team deliverable do not change
Finished Work
Sprint Retrospective

Figure 1. The Flow of Scrum Process (Sutherland and Schwaber 2007: 22)

Deleted: ¶

In other words the Scrum process begins with a vision of the system and release milestones. The vision is described in business terms rather than technical terms. The vision may be unclear at first but will become more precise as the project moves forward. The Product Owner is responsible for getting initial funding and creating the Product Backlog. The prioritized items in the Product Backlog are divided into smaller tasks through the Sprint Planning Meeting and placed in the Sprint Backlog. In the Sprint Planning Meeting, the Product Owner explains the content, purpose, meaning, and intentions of each item in the Product Backlog. All the tasks in the Sprint Backlog are done through the iteration of the Sprint which consists of the Daily Scrum Meetings (Cho 2008).

The daily meetings keep stakeholders up-to-date, and the planning meetings reduced the confusion about what should be developed (Dyba and Dingsøyr 2008; Mann and Maurer 2005). Scrum does not attempt to tackle ‘upstream’ activities such as project feasibility but it does cover the day-to-day project management responsibilities and general project coordination. However the success of Scrum development is heavily dependent on cooperation and collaboration among members of the development team and with the customer, as well as proactive accommodation of last minute changes (Kim 2007).

3.0 Research Methods

This research study's philosophy reflects the principles of the interpretivist or phenomenological (Blumberg *et al.* 2005) perspective and adopts a qualitative method of data gathering through a case study. The aim was to achieve a rich and detailed description of events, situations and the interaction between stakeholders and the events as they occurred (Cooper and Schindler 2006). The purpose is to obtain multiple perspectives of a single organisation over a period of time. Investigating an issue in more than one context is usually better than basing results on just one case as their results are considered more robust. However when a single case study affords access to information that is rarely accessible to researchers, a single case study is sufficient as it will offer as yet unknown insights – '*a single well-designed case study can provide a major challenge to a theory*' (Blumberg *et al.* 2005: 131). Additionally, *a single organisation does not necessarily prevent generalizability as this can take the form of concepts, theories, specific implications or rich insights* (Walsham 2006: 322). This case study opportunity enabled the researcher to contribute to the limited academic reporting of the views and opinions expressed in empirical settings (Dyba and Dingsøyr 2008) and address the lack of academic discussion in this area.

For this research study an *exploratory study* has been chosen as a valuable means of finding out *'what is happening; to seek new insight; to ask questions and to assess phenomena in a new light'* (Robson 2002: 59). The first step involved a search of the secondary literature. The secondary data was amplified with further reading of books, academic papers, conference papers and journals, as well as the examination of project documents and artefacts aimed at validating the data to strengthen conclusions drawn.

The primary research period lasted 2 years within the project environment and the data collection techniques employed involved observation, face-to-face interviews, online surveys, tele-conferences and spontaneous conversations. The principles of triangulation, awareness of contrariety and iterative discourse of data from different sources and time intervals were used to ensure rigour was applied to the analysis (Klein and Myers 1999). QSR NUD*IST Vivo (NVivo), a qualitative data analysis software tool was utilized to produce a database to store, manage and interrogate the data collected.

3.1 Observation

The initial 12 months of the research focused on the systematic non-participant observation. The purpose of non-participant observation was to observe and record what people do in terms of their actions and their behaviour at the time they occur in the natural setting without the researcher being involved (Cooper and Schindler 2006). In particular, the observational focus concentrated on daily Scrum stand-up-meetings, Scrum of Scrums, the Sprint Reviews and Sprint Planning meetings.

3.2 Interviews

After the observation phase came to an end particular employees specific to the case study research were asked to participate in one-to-one, informal semi-structured interviews in which anonymity and confidentiality were guaranteed (Walsham 2006). The response rate was high, 13 out of the 15 approached agreed to scheduled interviews. In particular 9 of the interviews were face-to-face, 1 was conducted over the phone and 2 via email. Interviews lasted between 45 minutes and 1 hour's duration to provide an in-depth and rich collection of detail of the case study project.

All interviews were audio-taped to record what participants said and the way in which they said it (Cooper and Schindler 2006). This was complemented with brief note taking to capture facial expressions and other non-verbal cues (Saunders *et al.* 2007). All taped interviews were transcribed and returned to interviewees for validation and to offset unintentional bias such that participants could clarify, delete or amend inaccurate or sensitive data.

The researcher chose the purposive sampling technique where interviewees were selected for their unique experiences, attitudes, or perceptions relevant to the case study project.

3.3 Spontaneous Conversations

Preece *et al.* (1998) recognize spontaneous conversations as being as important as formal communication as it represents an important aspect of the work environment. Therefore, this source of primary data offered an opportunity for further information to be collected but more significantly for clarifying issues that were not well understood during the meetings. All data was then directly imported into the qualitative software NVivo, along with electronic versions of any supporting documents, where it was organised, managed and subsequently coded for iterations of analysis.

3.4 Limitations

This case study research was subject to some limitations. For example, the neutral observation carried out in the project environment did not mean it was not biased by our own background and prejudices to see things in a certain way. Transcribing interviews and extracting themes proved to be very time-consuming that could have more effectively utilized but was crucial to the analysis phase. Tape-recording did not capture the non-verbal expressions and assessing human cognitions could be biased too. Also, tape recording may have influenced interviewees to be less honest (Walsham 2006). Although confidentiality and anonymity were guaranteed, when we referred to interviewees by their position we realized that this was not adequate to afford anonymity to everyone as some positions are quite distinctive. Therefore participants are referred to as ‘team members, senior members, participant and so on’ to prevent identification of views and opinions expressed.

Finally, the findings included in this paper consist of one single project within the organisation thus more studies of IS projects within the case study setting would further validate the analyses made. Indeed greater validity could be offered if the critical success factors were applied to a similar IS development project in a different organisation.

4.0 Case Study Context

The host organisation relevant to this research study is a medium-sized International Educational Foundation which has approximately 400 employees and 12 offices worldwide. It is a not-for-profit organisation; hence, any surplus generated is reinvested back into the organisation. This organisation has an internal Information and Communication Technologies (ICT) department and software is written in-house by the development team. Historically software development projects were based on the waterfall methodology. However, a decision was made to pilot test the agile approach of Scrum because of problems experienced with the waterfall approach of recent projects such as project overruns caused by its time consuming nature.

The analysis of the pilot project proved to be so successful that the organisation adopted Scrum immediately as a systems development approach. Consequently from December 2007 until April 2009 the host organisation gradually formed 5 Scrum Teams handling hundreds of small fixes, amendments and adjustments of previous projects which were not completed yet. For example, the Universities and Transcripts project of the research case study.

Scrum Team X was typically characteristic of what is expected. It consisted of a Project Manager, a Project Owner, a Scrum Master, one Senior Analyst Programmer, a Developer, Testers and Computer Programmers. End-users were represented demographically across Australia, Geneva, New York, Singapore, UK, and Vancouver. Other people less involved i.e. not on a daily or weekly basis include the ICT Development Manager, the ICT Director Assistant and the ICT Planning Controller.

It is not the remit of this paper to address the activities of the preceding traditional waterfall approach utilized for requirements elicitation or to document the transition but rather to concentrate on the Universities and Transcripts Project that is the focus of the research study.

4.1 The Universities and Transcripts Case Study Project

The Universities and Transcripts Project is a global software development started in July 2007 underwent a transition from the former used for the requirements gathering purposes to adopting Scrum for the development activities utilizing Scrum Team X ending in March 2009. This project consists of phase 1 and phase 2 deliverables made up of different components. This project has a high profile with external stakeholders of the organisation and needs to be compatible with and accessible by Universities and Admissions Body systems. The aim is to replace the client university and transcript system with one that is an integral part of the International Educational Foundation’s information systems which fully meets the requirements of all project stakeholders, where stakeholders include internal staff and all overseas offices during phase 1 and the universities in phase 2.

The project took a multi-phased approach. Phase 1 concerned gathering and collating the system requirements. It started in July 2007 and was completed in January 2008 as initially planned. It consisted of producing a proposed requirements specification that was circulated to the internal (i.e. UK) and external global stakeholders of the host organisation such that the new system would meet the users and business requirements. This document described both the legacy system as well as a description of the planned replacement.

Phase 2 started immediately after Phase1 and was due to be completed in December 2008 but was delayed until April 2009. This Phase dealt with the execution of the project. It consisted of simplifying the system by merging the many disparate legacy systems, the organisation NET, staff intranet and universities and transcripts system onto the host organisations IS. The administration side of the system that the universities and transcripts clients interact with is where they log in and view curriculum documentation, records and results across different file formats that they can then upload to their own systems. Specifically, the processing of transcript requests would be more efficient via the web page and comply with the new security measures. The user interface for both external and internal users of the system had to be improved and standardised due to the slow speed and lack of user friendliness.

Although the project was completed in principle it is considered a failure due to the overrun and the inability to meet the end-users' requirements. This paper examines the factors involved. We particularly examine the agility in terms of the values and principles of 'people, working software, end-user involvement and responding to change' as set out by Agile Alliance Manifesto (Beck *et al.* 2001).

5.0 Case Study Analysis

In this section we examine how the Universities and Transcripts Case Study Project managed the transition from the former waterfall structured approach to the agile Scrum approach and why the project is considered a failure. The following sections look at agility in terms of the values and principles of people, working software, end-user involvement and responding to change as documented by the Agile Alliance Manifesto (Beck *et al.* 2001).

5.1 People Are Important

This section examines problems experienced with the stakeholders firstly from an effective communications perspective, and secondly from the perspective of team dynamics.

5.1.1 People - Communication

The literature advocates face-to-face communications as it engenders a better level of understanding (Cockburn 2002). While video- and tele-conferencing and emails were utilized as channels of communication with the globally dispersed end-users, it was recognized that this was not the best medium. Indeed one team member agrees that '*…face-to-face communication is a lot easier'* but as this was not feasible other channels were used that were problematic. S/he reports that '*By the time we had sent regional offices an email, we had to wait a few hours before the office would get back to us and this was a drawback.'*

More seriously the geographical spread together with the diversity of working cultures meant that it was often difficult to get consensus. For example one participant reports

'…*not everyone in the regional offices worked the same way as UK does and they do some odd things to suit their own way of working. Australia might not like something that America wants so keeping all of them happy is not easy*'. This is made more difficult because each of the other offices does not have an understanding of the others who all have their own requirements.

An important aspect of communications is the ability of a global and diverse set of stakeholders to communicate effectively between each other, the developers and the host organisation. Although members of Scrum Team X recognize the importance of collaborative communications they reported '…*communication and dealing with people is the most difficult part, yet, the completion of the project depends on this*'. As the case unfolds we shall see how stakeholders found it difficult to communicate as part of a team that ultimately affected the success of the project.

5.1.2. People –Team Dynamics

For the case study it was not just a simple issue of poor communication skills but that some members of the Scrum team had difficulties in making the transition from the waterfalls prescriptive working patters to the agile feature of collaborative team working. Agility places a premium on people and their interactions. The emphasis is on teams and on the intense dynamics of team interactions rather than the individual developer as in traditional methodologies (Orr 2002). A senior manager agrees '*With agile it is so important that the team pulls together…we haven't done that adequately enough. In some instances … if a particular person is never going to be a team player take them out.'* In this way some benefits of an agile approach were lost to the project and we surmise that more effective team leadership could have prevented or limited this happening.

Observations confirm that the lack of team cohesion resulted in situations of conflict in task activities between the team members. For example a senior team member stated that "*Team working can be really difficult because we did have a conflict within one of the teams which made the entire team de-motivated'.* Indeed the literature supports this analysis. Organisations that migrate from a traditional methodology to an agile methodology can be subject to task conflicts (Balijepally *et* *al.* 2006; Nerur *et al.* 2005). For example, substantive issues such as differences in ideas, opinions, and ways of doing something can be problematic; and also relationship conflicts occur where interpersonal or socio-emotional disagreements generate feelings of animosity or annoyance (Bono *et al.* 2002). Another senior team member explains one such example *'I think that the particular situation we had was that we had two very strong personalities in the team who both made the other one feel that they were not being listened to which made both of them clash.'* A decision to change the team dynamics was implemented in an attempt to resolve this matter. A senior team member comments '*We chose to rearrange the scrum teams a few months ago because we felt the dynamics of this particular team were not working particularly well …* and continues '*Both very nice people and working very well with everybody else but it was just putting them together which was a problem. The best thing we could do was just take that team apart.'* This suggests that team personalities were a significant factor in achieving success for this research case study.

Observations confirm that the Scrum team did swap members to resolve conflict. However, this appeared to have had both a negative and positive effect. Interestingly three members of Scrum Team X reported negative views about changing team dynamics. For example one reports that *'When the team changes…it can be difficult because we then spend and absorb a lot of time supporting these* [new members] *because we all know each other's strengths and weaknesses and we know our roles within the team'*. A second acknowledges the same negative impact '*…we have lost people with valuable knowledge…but from the work point of view as well because the people that have come into the team they always have to get up to speed and to get to know the system*'. The third adds '*The negative side is people coming in and out which is a disruption to the team and with a lack of knowledge we have got to train them up…* However, a positive comment puts forward a different view '*Having people coming in and out of the team helps in sharing knowledge and we don't have to rely on one or two people because there is more people who will know about the system…* This was also acknowledged by other team members '*…the positive side is that more people have a widespread knowledge of all these areas.'*

Even though literature also supports the positive aspect that role changes foster understanding of the system and support the notion of collective responsibility promoting greater knowledge sharing (Balijepally *et al.* 2006) in reality the collective interpretation of the Scrum Team members may experience more negative impacts.

5.2 Working Software

In agile methods, working software is demonstrated to end-users early and often through short development iterations and is more highly valued than comprehensive documentation. The motivation for this value is the belief that '*A better way to ensure customer needs are met is through working software rather than through formal written words*' (McMahon 2005: 1). However, when the Programmers were asked to talk about working software it became obvious that the third principle of the Agile Manifesto was not applied as expected. For example they reported *"The emphasis here is that we tend to emphasize upfront analysis and documentation rather than delivering software to customers every 3 weeks. We certainly don't do that; I can't remember the last time we did a product demonstration'.* In fact we propose the process implemented reflected a more structured approach like the waterfall model i.e. analysis, documentation and then development and testing. Programmers agree with this analysis and continue '*We don't do a demonstration in each sprint, where we could be receiving feedback about the product or what the customers want. We only deliver software after a whole set of functionality is complete so we end up doing a waterfall approach but doing it every three weeks.*'

Other members of the Scrum team were also aware of the lack of working software demonstrated, for example '*In the last few months we have not really demonstrated a system to be honest I don't really remember when it was the last time that we demonstrated some work';* others put forward similar views *'I could probably count on one hand the amount of product demonstrations we have done'.* From this we analyse that although Scrum practices were evident in reality they were not always followed or applied in their entirety, previous inherent waterfall practices took over.

Although, agile methodologies are associated with minimal documentation (Boehm 2002; Highsmith 2003) data gathered through observations and from interviews suggests the opposite is may be true for this research project. Scrum Team X's developers maintain that there was an emphasis on producing documentation that was felt to be both excessive and onerous. They report '*…the document is still far, far too excessive sometimes we write documents such as this goes here and they actually write the components before you know you are writing the whole program in a word document yet again which is absolutely absurd.'*

A more positive view states that '*We have changed from lengthy project and development specifications to just having some small UML documentation. There isn't the full planning documentation as one would expect from a more traditional project but there are some pieces of information such as the original outline form, and the Scrum documentation such as the Product Backlog.* This clearly differs from the developers' experience. There is evidence to suggest that it is the interpretation of what 'just enough or barely sufficient' documentation may be (Cockburn 2002). However one team member goes on to clarify that '*We need to ensure that all the information is documented enough to enable other people to pick that up and be able to continue to work. This means that we also need to go back over our old style documentation and update it to the new style'* and this differs notably from the previous statement of '*some small UML documentation'*. It can be argued that as this team member is less involved their view may not be as informed as that of the team member who has a daily involvement with the project. Thus the views and opinions expressed reflect the extent to which people were subjectively involved.

From the above analysis we can surmise that perhaps not all the Scrum team members were actively receptive of the new agile methodology. We would also reaffirm our earlier analysis that team management of this project was poor and insufficient to deal with the divide in the Scrum team, a team that went through a number of revisions which in itself suggests a lack of control.

5.3 End-User Involvement

This section examines the third principle of the agile manifesto - that of customer collaboration over contract negotiation. Therefore it is necessary for the business people and the developers to work closely together supported by regular customer feedback throughout the project duration. As stated previously end-users were demographically represented across the global context of Australia, Geneva, New

York, Singapore, UK and Vancouver. This geographical spread meant that co-location of end-users for the project duration was not feasible. Consequently the end-user attending the 3 weekly sprint cycle was UK located.

The Project Manager confirms that the end-user involvement was achieved singularly through a senior business manager located in the UK who also acted as a subject expert and a representative of all global end-users. The Project Manager reported that '*Nobody else was directly involved in the project. Everything was channelled through this person because she had the expertise to answer the questions to the development team'*. Observations and interviews support this analysis. For example, the senior business manager involved confirms *'Only I was present at the end of sprint meetings because I was the only end-user in the* [local] *office'*. She feels that this was particularly advantageous reflecting '*I don't think I would change that because our interaction worked quite well, it's good to be open and I would rather them ask me the questions than not ask me'*. This was supported by occasional tele-conferences between the development team with the other global stakeholders.

However it must be remembered that the system requirements had previously been gathered through an initial upfront structured waterfall approach during which intensive consultation had occurred with the globally dispersed end-users. A systems specification document was produced and signed off by these stakeholders. A tester further clarifies that globally dispersed end-users *'…do get involved during the Sprint, not in the daily Scrum meetings or Demonstrations but they have access to the code so they actually run the functionality on their own PCs so they can give us feedback…'* Contrarily to this belief one of the global end-users from Vancouver felt insufficient time was allocated to them to do testing before the site was launched '*Often we had not seen the mock-ups and were not able to fully test a particular section of the site before it was deemed 'completed'*. It was felt that their feedback was not always taken on board.

Another distanced end-user in Singapore experienced similar problems reporting that emails on development issues were often sent in large clusters that proved difficult to track and deal with. He reports that '*It was as if all of the stakeholders were expected to drop everything while the 'sprint' was taking place.'* What is evident here is that when the project shifted from the waterfall approach to Scrum the development practices and accompanying activities changed radically. The level of interaction with stakeholders increased considerably such that *'There was constant request for information… involvement gets so much that perhaps it prevents work being done'*. A senior team member reports that as a consequence when we requested feedback it was not always forthcoming at the time it was needed. This had a negative effect on the development team and caused difficulties between them and the globally dispersed end-users.

5.4 Responding to Change

Since change is inevitable and often unpredicted in software development then all stakeholders, (here we mean the people affected by the change as in the development team, business people, IT department, clients and so on) must be able to accept, react and welcome it even if late in the development process (Koskela 2003; Pressman 2001). However it is not always a matter of the change itself but sometimes has more to do with the processes utilized to manage 'that' change issue – this is the situation with the research case study. For example the need to respond rapidly to change in the current dynamic business environments was recognized compared to the traditional approach of developing a costly system over a number of years. For example '*You have to change the methodology to fit the way the world now is. The world now is moving much quicker, businesses have to be much more reactive and move much more quickly…'* Further explanation confirms that the move to agile development would allow the host organisation to be significantly more responsive to the business changes particularly while they were reorganizing and restructuring the business that was not achievable using a waterfall methodology.

A commonly held view was that business prioritized the delivery schedule ahead of the development approach. They were more concerned with getting the end product delivered on time rather than the process being applied by the ICT department. For example one common belief voiced was '*The business is quite happy if we said we were going to use some other method as long as they got the product they wanted.'* This significant point epitomizes the extent of this disinterest '*One of director said to me 'I really don't want to hear about agile anymore, talk to me about products that* *you are delivering, don't talk to me about framework.'* Hence this personal view reflects how this ICT attitude affects the perception of ICT *'ICT is considered to be one of those technical professions that go on in the back… However, a lot of it when you talk about ICT, you see the shutters go down.'* This clearly illustrated a lack of commitment to change.

Indeed it is the lack of response to changes that is 'blamed' for the overrun of the project causing it to be viewed as a failure. Problems were experienced with end-users not knowing what they want *'A number of times we worked or delivered a solution to them and they said well it is not quite right we want…it was only when we implemented something that they went away… and then they said oh well we want these changes to be done...we changed our mind'.* Thus, numerous changes raised in the sprint meetings would have traditionally been considered as change requests and been dealt with accordingly. However with a Scrum approach even though accommodating the end-users' changes means that the system more closely meets the customers' requirements the extra work involved is at the expense of the delivery deadlines and schedules. A manager comments *'…there were occasions that sprints were not completed and a number of hours of work carried forward… this extra work feeds into the scrum so for the universities we were expecting to finish December 2009 but we ended up finishing in April 2009 because there some pieces of work that the end-user required.'*

This is an age old problem and one that agile development is theorized to accommodate through prototyping and demonstrating working software. However, as discussed in the above section the amount of software demonstration conducted falls far short of that expected with an agile project. However, the problems being experienced were more representative of a waterfall approach. Problems were being raised at implementation after months of development work rather than during the demonstrations iteratively as is the case with agile development. More evidence of agility may have prevented the backlog of changes that pushed the system into overrun, and failure.

6.0 Conclusions

Through the lenses of an IS development project we have examined the agile values and principles of people, working software, end-user involvement and responding to change of an IS development project that underwent a transition from the waterfall approach to a Scrum agile approach during development.

The case study illustrates that although the host organisation moved from a waterfall methodology to Scrum agility it was not wholly embraced. Although evidence confirms that the main stages of Scrum development occurred the development activities were in part driven from a more waterfall emphasis. Indeed the implementation reflected the more structure approach of a waterfall model i.e. analysis, documentation and then development and testing rather than the anticipated Sprints. More emphasis is needed on preparing the environment by training the stakeholders in agile behaviour from both a process standpoint and a 'thinking/attitudinal' perspective (Dyba and Dingsøyr 2008; Mann and Maurer 2005; Nerur *et al.* 2005). The case clearly underlines the need to optimize and harmonize the structure of Scrum teams that epitomize particular roles. Where there is conflict between the members then problems will follow as evidenced in the case study. The Scrum team underwent a number of changes to deal with such conflict.

Collective consensus was affected by the multicultural interaction between developers and clients together that was further compounded by time delays and the inability to maintain an awareness of working context at remote sites that all contributed to major challenges affecting the entire software development process. Failure to fully understand the required system features, and the inability to effectively resolve conflicts resulted in budget and schedule overruns effecting failure (Damian and Zowghi 2003). Potentially, for this case study where on-site customer collaboration and face-to-face interaction is severely reduced then such agile methodologies as Scrum are more difficult to achieve.

However, as Coughlan and Macredie (2002) maintain the diffusion and adoption of a system development approach is impacted by the absolute nature of the host organisation. A methodology does not necessarily map directly onto an understanding of the organisation, its rationality or the context of its users. The move from the previous traditional development culture affected the anticipated agility of the Scrum approach (Highsmith 2000, 2002; McConnell 1996) which proved problematic for this case study. Set out below is a set of critical factors to be considered when making the transition from traditional to agile development approaches aimed at informing future development practice.

Traditional to Scrum Agile Development Critical Factors
Preparation of the Environment (i.e. adequate Scrum Training)
Effective Communications
Optimal Team Structure (i.e. personalities)
Effective Team Leadership

Table 1. A Set of Critical Factors for Scrum Development

In practice more emphasis is needed on preparing the environment by training the stakeholders in agile behaviour from both a process perspective and the adoption of a common mindset. Team structure needs to be optimized that epitomizes the particular roles required with Scrum development. Effective communications will facilitate interaction and collective consensus by the different multi-cultural stakeholders. Finally effective leadership across the global context although difficult to accomplish is desirable to achieve effective cooperation and collaboration.

References

Abrahamsson, P., Salo, O., Ronkainen, J. and Warsta, J. (2002) Agile software Development Methods: Review and Analysis, *Espoo VTT Publications* 478 [Available URL: http://www.pss-europe.com/P478.pdf] [Accessed on: 15th March 2008].

Balijepally, V., Mahapatra, R. and Nerur, S. (2006) Assessing Personality Profiles of Software Developers in Agile Development Teams, *Communications of AIS*, Vol. 18, pp. 2-40.

Beck, K., Grenning, J., Martin, C.R., Beedle, M., Highsmith, J. and Mellor, S. (2001) *Manifesto for Agile Software Development*,

[Available URL: http://agilemanifesto.org/]
[Accessed on: 30th September 2007].

Blumberg, B. and Cooper, D. R. and Schindler, P. S., (2005) *Business Research methods*, McGraw Hill Education, Berkshire.

Boehm, B. (1999) Making RAD work for your Project, *IEEE Computer*, Vol. 3, pp. 113-117.

Boehm, B. (2002) *Get Ready for Agile Methods with Care*
[Available URL: http://www2.umassd.edu/swpi/xp/papers/r1064.pdf]
[Accessed on: 25th January 2007].

Boehm, B. and Turner, R. (2003) Using risk to balance agile and plan driven methods, *IEEE Computer*, Vol. 36(6), pp. 57-66. [Available URL: http://www.acq.osd.mil/se/as/publications/IEEE%20Software%206-03.pdf]
[Accessed on: 13th October 2008].

Bono, J. E., Boles. T.L., Judge, T.A. and Lauver, K.J. (2002) The Role of Personality in Task and Relationship Conflict, *Journal of Personality*, Vol. 70(2), pp. 311-344.

Cho, J. (2008) *Issues and Challenges of Agile Software Development with Scrum*, In Proceedings of 48th Annual IACIS International Conference, October 1-4, Savannah, Georgia, USA.
[Available from: http://www.iacis.org/iis/2008_iis/pdf/S2008_950.pdf].

Cohn, M. (2004) *User Stories Applied*, Addison-Wesley Professional.

Cockburn, A. (2000) *Writing Effective Use Cases-The Crystal Collection for Software Professionals*, Addison-Wesley, Reading, MA.

Cockburn, A. and Highsmith, J. (2001) Agile software development: The business of innovation, *IEEE Computer*, Vol. 34(9), pp. 120–122.

Cockburn, A. (2002) *Agile software development*, Addison-Wesley, Reading, MA.

Cooper, D. R. and Schindler, P.S. (2006) *Business Research Methods*, (9th edn), McGraw-Hill, New York.

Coughlan, J. and Macredie, R. D. (2002) Effective Communications in Requirements Elicitation: a Comparison of Methodologies, *Requirements Engineering*, Vol. 7(2), pp. 47-60.

Damian, D. and Zowghi, D. (2003) Requirements Engineering Challenges in Multi-site Software Development Organizations, *Requirements Engineering*, Vol. 8, pp. 149-160.

Dyba, T. and Dingsøyr, T. (2008) Empirical studies of agile software development: A systematic review, *Information and Software Technology*, Vol. 50, pp. 833–859.

Elliott, E. (1997) *Rapid Applications Development (RAD): an odyssey of information systems methods, tools and techniques*. 4th Financial Information Systems Conference, Sheffield Hallam University, U.K.

Erickson, J., Lyytinen, K. and Siau, K. (2005) Agile Modeling, Agile Software Development and Extreme Programming: The State of Research, *Journal of Database Management*, Vol. 16(4), pp. 88–100.

Graham, D. R. (1989) Incremental Development: review of non-monolithic lifecycle development models, *Information and Technology Software*, Vol. 31, pp. 7-20.

Highsmith, J. (2000) Retiring Lifecycles Dinosaurs, *Software Testing & Quality Engineering*, July/August, pp. 22-28.

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

Highsmith, J. (2003) Cutter Consortium Reports: Agile Project Management:

Principles and Tools, *Cutter Consortium*, Vol. 4(2), Arlington, MA.

Iivari, J., Hirschheim, R. and Klein, H.K. (2000) *A Dynamic Framework for Classifying Information Systems Development Methodologies and Approaches,* Journal of Management Information Systems, Vol. 17(3), pp. 179-218.

Kim, Y. (2007) *Analyzing Scrum Agile Software Development with Development Process, Social Factor and Project Management Lenses*, In Proceedings of the 13th Americas Conference of Information System *(AMCIS-07).*

Koskela, J. (2003) Software configuration management in agile methods, *Espoo VTT Publications.* [Available URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.106.4156&rep=rep1&type=pdf]
[Accessed on: 23rd February 2008].

Klein, H.K. and Myers, M.D. (1999) A set of Principles for Conducting and Evaluating Interpretative Field Studies in IS, *MIS Quarterly*, Vol. 23(1), pp. 67-94.

Kruchten, P. (2000) *The Rational Unified Process: An Introduction*, Addison-Wesley, Reading, MA.

Leffingwell, D. (2007) *Scaling Software Agility-Best Practices for Large Enterprises*, Addison Wesley, Boston, MA.

Lindstrom L. and Jeffries R. (2004), Extreme programming and agile software development methodologies, *Information Systems Management,* Vol. 21(3), pp. 41–61.

Lindvall, M., Basili, VR., Boehm, B., Costa, P., Dangle, K., Shull, F., Tesoriero, R.,Williams, L., Zelkowitz, M.V. (2002) *Empirical findings in agile methods Extreme Programming and Agile Methods*—XP/Agile Universe 2002, Springer, Berlin.

Mann, C. and F. Maurer (2005) *A Case Study on the Impact of Scrum on Overtime and Customer Satisfaction,* In Proceedings of XP/Agile Universe 2005, Denver Colorado, US.

Martin, J. (1991) *Rapid Application Development*, New York: Macmillan.

Martin, R. C. (2003). *Agile software development: Principles, patterns, and practices*. Upper Saddle River, NJ: Prentice Hall.

McAvoy, J. and Butler, T. (2007) The impact of the Abilene Paradox on double-loop learning in an agile team, *Information and Software Technology*, Vol. 49, pp. 552-563.

McConnell, S. (1996) *Rapid Development – Taming Wild Software Schedules,* Washington: Microsoft Press.

McMahon, P. E. (2005) *Extending Agile Methods: A Distributed Project and Organisational Improvement Perspective*
[Accessed on: 3rd February 2008] [Available ULR: http://www.sstc-online.org/Proceedings/2005/PDFFiles/PEM786pap.pdf].

Nerur, S., Radhakanta, M. and Mangalaraj, G. (2005) Challenges of Migrating to Agile Methodologies, *Communications of the ACM*, May2005, Vol. 48(5), pp. 73-78.

Orr, K. (2002) CMM versus Agile Development: Religious Wars and Software Development, *Agile Project Management Advisory Service*, Vol. (3)7, pp. 29.

Preece, J. Rogers, Y. Sharp, H. Benyon, D. Holland, S. and Carey, T. (1998) *Human-Computer Interaction*, England: Addison Wesley.

Pressman, S.R. (2001) *Software Engineering: A Practitioner's Approach*, (5th edn), McGraw-Hill, New York.

Raffoni, M. (2000) Got a need for Speed, what you can learn from RAD, *Harvard Management Update*, Vol. 5(11), pp. 10.

Robson, C. (2002) *Real World Research*, (2nd edn), Blackwell, Oxford.

Sanders, M., Lewis, P. and Thornhill, A. (2007) *Research Methods for Business Students*, (4th edn), FT Prentice Hall, Harlow, England.

Schwaber, K. and Beedle, M. (2002) *Agile software development with Scrum*, FT Prentice Hall, Upper Saddle River, NJ.

Stazinger, J. W., Jackson, R. B. and Burd, S. D. (2005) *Object-oriented analysis and design with unified process,* Thomson Course- Technology, Boston.

Stapleton, J. (1997) *Dynamic Systems Development Method – The method in practice,* Addison Wesley, Harlow, UK.

Sutherland, J. and Schwaber, K. (2007) *The Scrum Papers: Nuts, Bolts and Origins of an Agile method,* [Available URL: http://scrumtraininginstitute.com/home/stream_download/scrumpapers] [Accessed on: 18th July 2008]

Sutherland, J. Jacobson, C. and Johnson, K. (2007) Scrum and CMMI Level 5: A Magic Potion for Code Warriors!, in *Agile 2007*, Washington, D.C.

Takeuchi H. and Nonaka, I. (1986) The New Product Development Game, *Harvard Business Review,* Vol. 64, pp.137-46.

Vinekar, V., Slinkman, C. W. and Nerur, S. (2006) Can Agile and Traditional Systems Development Approaches Coexist? An Ambidextrous View, *Information Systems Management,* Summer 2006, Vol. 23(3), pp. 12, 31-42.

Walsham, G. (2006) Doing interpretive research, *European Journal of Information Systems*, 15(3), 320-330.

Williams, L. and Cockburn, A. (2003) Agile Software Development: it's about Feedback and Change', *IEEE Computer,* Vol. 36(6), pp. 39-43. [Available URL: http://csdl2.computer.org] [Accessed on: 10th October 2006]