UNIVERSITY OF SOUTHAMPTON

Faculty of Physical and Applied Sciences

Electronics and Computer Science

Electronic and Software Systems

Nine-month progress report

Toward a Framework for Localisation of Product Software across Organisational Boundaries

Supervisor: Dr Gary B. Wills

Supervisor: Dr Andrew M. Gravell

Internal examiner: Dr Robert J. Walters

By

Abdulrahman Mohammed Qahtani

February 20, 2012

Abstract

Distributed agile development (DAD) is a current trend for software development. It uses agile practices to promote iteration and flexibility in the distributed development of software projects. DAD involves a software vendor and their customers working together, leading to an overlap between their organisations. In this report, which is a progress report submitted for continuation towards a PhD, we introduce the agile software development and propose a framework for the localisation of software products across organisational and cultural boundaries. The framework addresses and accommodates the key components of the area between software vendors and customers. Our approach is useful in that it helps project managers, stakeholders and developers to understand the correlations and critical factors associated with customers and software vendors. This framework tries to cover all the important aspects of the development of agile software across distributed organisational cultures instead of focusing on a specific aspect such as project management.

Table of Contents

Chapter 1 . Introduction

Agile software development is a significant departure from the plan-based approaches of software engineering (Morien and Wongthongtham, 2008). The issue of how software products can be produced and delivered faster, better and cheaper is the main motivation of the huge demand to adopt agile in different software projects. As a matter of fact, agile methods have promoted iterative approach principles as well as agile values to meet that demand for producing faster software products (Abrahamsson et al., 2002).

On the other hand, software producers are looking at lower costs and highly skilled human resources to develop software products. Thus, the concept of distributed software development (DSD) has appeared. Although there are several advantages to this concept, there are disadvantages such as communications challenges, the cultural difference issue and the difference in time zones (Jiménez et al., 2009). Over the last two decades, agile methods have been adopted on a number of occasions, as well as distributed software development in different sized projects (Beck, 1999). Consequently, the new trend in agile adoption is to apply agile principles to DSD projects to achieve the features of DSD and agile methods at the same time. Adopting agile methods on DSD often increases some of the challenges of DSD, such as communication, due to the emphasis of the agile approach on face to face communication (Fowler and Highsmith, 2001) which does not exist in DSD.

Despite this fact, several distributed agile development projects have been successful in the industrial context (Sureshchandra and Shrinivasavadhani, 2008). The current PhD focuses on the organisational boundaries between software producers and stakeholders. In view of the challenges and issues that face the adoption of agile distributed software development in order to deliver and localise software, there is a lack of suitable frameworks for localising software products across organisational boundaries to ensure success in the development and localisation process by using agile and traditional methods, and thus achieve customer satisfaction. To address this gap, we introduce the agile approach in a particular scenario, as well as proposing a framework to accommodate the key aspects of organisational boundaries that should be considered during the development and localisation process.

The rest of the report is structured as follows. In Chapter 2, the distributed agile development background and a literature review of frameworks and models proposed for DSD and DAD. In addition, we present some research discussing the issues and challenges of development across distributed projects. In Chapter 3, we discuss the introduction of agile software development in organisational boundaries and the proposed framework. In Chapter 4, we provide a conclusion of the report, followed by research questions and future work.

Chapter 2 . Review of Development and Localisation for Software Products across Boundaries

2.1 Introduction

The localisation of software products across organisational boundaries has many different related aspects and disciplines, such as software engineering and management. Thus, this chapter will discuss research and studies that have been conducted in terms of a proposed new framework and models in this particular area. It will also discuss the main factors that have an effect on the localisation of software products in a distributed environment, especially if there are different teams as well as different development approaches, such as traditional approaches like the waterfall model and agile software development methodology.

2.2 Agile software development background

Agile software development is “a phenomenon” (Dingsøyr et al., 2008) and not merely a development approach or methodology; it is actually a philosophy of software development and a new way of thinking in development process and project management (Shore and Warden, 2007; Fowler, 2001). It is the demand of the business community (Abrahamsson et al., 2002) to find a development method which would be lighter and faster than the traditional approach, plan-based models. It is a reaction against traditional models such as the waterfall model to reduce development time and costs, as well as to accommodate any change in requirements at any time without a significant effect on the whole development duration. As a result, the agile method was a sensation in the software development process and community (Cohen et al., 2004).

2.3 Agile Manifesto Review

In early 2001, seventeen agile practitioners and their proponents gathered in order to discuss the agile method. The main motivation behind that meeting was to strike a balance in the amount of modelling, documentation and planning in software development (Cohen et al., 2004). Since traditional methods emphasised those aspects, “the Manifesto has become an important piece of the Agile Movement” (Cohen et al., 2004) as it had representatives from different agile methods and technologies, such as Extreme Programming (XP), DSDM, SCRUM, Crystal, Feature-Driven Development, Adaptive Software Development, Pragmatic Programming and others (Fowler and Highsmith, 2001).

Also, Fowler and Highsmith (2001) say that "the Agile movement is not anti-methodology". The manifesto reads as follows (Beck et al., 2001):

We are uncovering better ways of developing software by doing it and helping others do it. Through this work, we have come to value:

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

That is, while there is value in the items on the right, we value the items on the left more.

The agile manifesto focuses on relationships, developers and the human role (Abrahamsson et al., 2002). Glass (2001) states that traditional software development places emphasis on process more than people, although the practitioners notice that people matter in software development. The second value of the agile manifesto is less emphasis on documentation which is agreed by the agile community (Glass, 2001; Abrahamsson et al., 2002). The balancing on documentation over years and levels is required but the main emphasis should be on producing working software as an ultimate product. The third and fourth values are a focus on flexibility in requirements changes and collaboration with customers in order to gain customer satisfaction and reduce the cost and time of development. Furthermore, the agile manifesto makes a collection of twelve principles beside those four values (Fowler and Highsmith, 2001). These values and principles together in practice would be the best way to be agile (Shore and Warden, 2007).

2.4 Distributed Agile Development background

Distributed agile development (DAD) refers to adopting agile principles in distributed software development (DSD) to achieve the features of agile software development and the advantages of using distributed development projects. As agile practices promote the development iteration process through agile methodologies, this can help DSD to tackle its challenges and issues, such as the difference in culture and communication (Phalnikar et al., 2009). However, there are many stories of organisations adopting agile methods in distributed development environments in different forms (Lee and Yong, 2009).

2.5 Review of Proposed Framework and Models in Terms of DSD and DAD

Little research has proposed frameworks and models to provide a guide for developers and managers in the agile development process for distributed projects.

Šmite and Borzovs (2006) conducted a study to investigate the impact of risk management on GSD, which is called global risk. In addition, they designed a framework to address the key risk management in global software development (GSD) across organisational and cultural boundaries. Wahyudin et al. (2008) proposed a framework for communication and information exchanges between development team members in GSD. This paper focused on the communication aspect in GSD with agile software and the notification of the development process. In addition, they proposed a concept to formalise the key communication between teams in agile projects to reduce the challenge of communication and the cost and to gain the benefit of communication in a distributed agile development (DAD). Akbar et al. (2008) proposed a model for those software companies developing web applications for distributed client locations. Their proposed model emphasises the support that is needed for communication between developers and offshore clients to complete their projects with the minimum documentation. Hossain et al. (2009) conducted a survey to investigate and identify the challenges of applying an agile method called Scrum on GSD. Furthermore, they proposed a conceptual framework that presents the key challenges of using Scrum in GSD projects. Their framework could help project managers who are using Scrum on GSD to consider the challenges and risks that could face their project in order to reduce them. Lee and Yong (2009) conducted a study that examined the main issues of global DSD and the challenges facing the distributed localisation teams of software products. Furthermore, they suggested a framework to map the challenges of project management in the globalisation of DAD to practices. Mudumba and Lee (2010) found that there was a lack of studies conducted on the risk management of DSD. As a result of this, they proposed an agile risk management framework that supported an identification process of dynamic risk management for DSD. Interestingly, they discussed multi-organisations, multi-teams and other multiplicities in DSD. In addition, they reported that several researchers had recommended this type of agile method in project management to mitigate the dynamic risk in software development projects. Phalnikar et al. (2009) carried out a study to investigate the benefits of using agile methodology like Scrum in distributed software development projects. They presented some of the challenges of DSD, such as communication, configuration management, project estimation and cultural challenges. In addition, they showed the benefits of agile distributed development. The scope of their study covered projects using a traditional development approach and agile adoption of those projects. Furthermore, they proposed two models for team structure in DSD.

Table 1 shows the frameworks and models that were proposed to address some of the challenges and issues of distributed software development and apply agile concepts on distributed software development projects. Furthermore, the table has the main contribution and discussed aspect of each study.

Table 1: Frameworks and model proposed for DSD and DAD

Paper	Contribution	Aspects	Research method for generation and evaluation of result
(Lee and Yong, 2009)	Framework	Project management and GSD	Generate their result and evaluated form case study on My Yahoo! 'Zorro' across 17th international contraries
(Šmite and Borzovs, 2006)	Framework	Risk Management of GSD	Conducted a case study in software houses in Latvia. In addition, a single case study was conducted in another company. There was no validating result. However, the research results were validated using a global and in-house project survey.
(Hossain et al., 2009)	Framework	Risks of applying SCRUM in GSD	Result is not evaluated
(Mudumba and O.-K. (Daniel) Lee, 2010)	Framework	Agile Risk Management of GSD	The proposed framework evaluated by reflects that framework on case from literature, which is the Skandia Financial Concepts (SFC) case.
(Phalnikar et al., 2009)	Model	Team structured in DSD projects	Result is not evaluated
(Akbar et al., 2008)	Model	Communication between people in DSD projects	Result is not evaluated
(Wahyudin et al., 2008)	Framework	Communication and exchange of information	This study used an initial empirical evaluation by using a scenario of requirements and then comparing the results with alternative requirements

2.6 Challenges and Issues of DSD and DAD Review

Coram and Bohner (2005) examined the impact of agile methods on software project management. Their study discussed the impact of applying agile in the project process as well as the people involved in the project process, such as developers, testers, project leaders and customers. Also, they discussed some management and development processes (e.g. planning and documentation). Some researchers have focused on one aspect of project management, such as risk management, and then examined the impact of distributed development or agile development on this aspect. Jiménez et al. (2009) conducted a systematic review of the literature relating to the challenges and issues of distributed software development. In addition, their study shows the proposed solutions and meeting of those challenges. They addressed the challenges of DSD projects, such as communication, group awareness (relationship between people in the project), configuration management, knowledge management, coordination, collaboration, project and process management, process support, risk management, cultural differences and quality measurement. In addition, they presented a proposed solution or way of meeting each challenge at that time. Sengupta et al. (2006) have done research initialled by study at the IBM research centre to investigate the challenges of DSD. They identified four areas in DSD, which are collaborative software tools, knowledge acquisition and management, testing in a distributed set-up and process and metrics issues.

In addition, they addressed the issues and difficulties of each area as well as presenting the research gaps for those areas, such as inadequate communication, trust, system integration and knowledge management. Damian and Zowghi (2007) investigated requirement engineering challenges and issues in distributed software development, especially across cultural boundaries and those existing in stockholder organisations. These authors have been able to construct a model on the requirement gathering process, including negotiation and specification. They show the difficulty of the development process in DSD projects in terms of requirements engineering. Fowler (2003) has written about his experience of adopting agile principles in an offshore development project. In this report, he discussed the importance of some factors in agile development (e.g. communication, cultural changes and documentation). In addition, he presents the challenges as well as benefits of applying agile in offshore projects. He also discusses the current and future trend of agile offshore development, stating “*Offshore development is very fashionable*”. Rodríguez et al. (2010) have conducted a study to investigate the tools and technologies that are used by distributed teams. They discussed the collaboration and integration of these technologies and the tools involved in software processes, such as IBM Jazz, Microsoft SharePoint and Google Apps. Their study included a comparison between these technologies and the benefits of tools and technologies in the software development process, like tracking systems, management features and calendar management. Table 2 shows a summary of the research that has been conducted to investigate the challenges and issues in different factors like communication, knowledge and requirements.

Table 2: Summary of research conducted on DSD and DAD with main discussed aspects.

Paper	Factors	Methods
(Coram and Bohner, 2005)	Project management, people, planning, documentation, development process.	Using the qualitative approach to generate the result. However, there is no evaluation.
(Jiménez et al., 2009)	Cultural differences, group awareness, configuration management, knowledge management, coordination, collaboration, project and process management, process support, risk management, quality and measurement.	Systematic literature review.
(Sengupta et al., 2006)	Collaborative software tools, knowledge acquisition and management, testing in DSD, process and metrics issue	Using initial case study in IBM. However, there is no evaluation for results.
(Damian and Zowghi, 2007)	Requirement engineering and its challenges in DSD, like technology, culture and informal communication.	Conducted case study in the Global Development Systems (GDS) company in the US to find results and evaluate them.
(Fowler, 2003)	Cultural changes, requirements, documentation, costs, project management and future of DSD.	There is no evaluation.
(Javier Portillo Rodríguez, Christof Ebert, 2010)	Discussed some collaborative technologies like IBM Jazz, Microsoft SharePoint, Google Apps and IBM Lotus	Comparison between some collaborative technologies.

2.7 Summary and Discussion of Literature Review chapter

The literature review helped us to understand the main idea behind distributed software development as well as adopting agile principles for DSD projects. In addition, it presented the previous work and research that was conducted on applying agile principles in DSD projects to understand the challenges and issues of this process and the key factors that would have an effect on the development process at organisational boundaries. We separated the previous literature into two sections. The first section researches and studies the proposed frameworks or models. We found some frameworks and models discussed the project management and risk management challenges in DSD and DAD projects (Lee and Yong, 2009; Šmite and Borzovs, 2006; Hossain et al., 2009; Mudumba and Lee, 2010). Other research proposed frameworks or models to cover communication aspects in terms of DSD as well as DAD (Akbar et al., 2008; Wahyudin et al., 2008). One piece of research has proposed two models for team structures in DAD projects (Phalnikar et al., 2009). Although many studies have proposed frameworks and models on different aspects like communication, project management and the team-structured challenges of DSD or DAD, there is a lack of frameworks which discuss all the key factors of the development process in DSD and organisational boundaries.

The second section presents research that includes case studies, systematic review or the investigation of DSD and DAD, along with studies which discuss the challenges and issues of development in distributed projects in general or in specific aspects, like project management, requirements engineering and communication. This research helps developers, project managers and stakeholders to consider the key factors and challenges of development in distributed projects.

Table 3 shows the factors and aspects that have been discussed in the two previous literature sections. While some of these aspects have been presented in the proposed frameworks or models in section one, such as project management, risk management and communication, no framework has been proposed to address all of these aspects in terms of organisational boundaries in distributed software development.

Table 3: Factors which have been discussed in the literature review.

Communication	Testing	Collaboration
Documentation	Tools and technologies	Group awareness
Project management	Requirements	Quality
Risk management	Time zone	Measurement
Configuration management	Knowledge management	Planning
Trust	Process management	People
Culture	Coordination	Integration

Chapter 3 . Framework for Localisation of Product Software across Organisational Boundaries

3.1 Introduction

In the previous chapter, we noticed that many aspects (Table 3) discussed, either in frameworks or as a review, the challenges and issues of the development process in DSD and DAD. However, no framework proposed to address all those aspects illustrated in Table 3. The purpose of this project was to investigate the key factors of localisation of product software across organisational boundaries and discuss the main challenges and issues. In addition, introducing the agile approach to distributed software development across organisational boundaries like that in Figure 1.

Organisational boundary is that area which comes from the overlap of multi-organisations. Some researchers define organisational boundaries as a central phenomenon viewed with multi theoretical lenses (Santos & Eisenhardt, 2005). In our research, there are two different types of organisational boundaries. The first type is an inter-organisational boundary that appears between a software producer and customers' organisations. The second type is an intra-organisational boundary that shows inside an organisation, such as boundaries between the localisation team and customers or management level and software development level (Figure 1).

3.2 Agile Approach across Organisational Boundaries

During a localisation process for any software product, there are new development requests required by the customer. There are two ways to meet these requests: either develop those requests in current version or in the next version of that software product. Those that will be developed in the next version would take at least six months and usually follow a traditional approach like waterfall. In this study, we will introduce an agile software development approach to develop requests and requirements on the current version. Figure 1 shows that the development team at a software producer site are divided into two stages. Stage one is developing new versions of that software product by traditional approaches such as the waterfall approach. Stage two is developing new features or classes based on a customer’s requirements of current versions in short term plans. Actually, there are many advantages to using agile principles in developing a customer’s requirements in a short term iteration process:

- By using agile at the customer's location, communicating requirements are easier.
- Applying the customer's requests in short iteration would make the localisation process faster and easier.
- The localisation team and customer working together in a small team will help to convey and exchange important information to promote the localisation and development process.

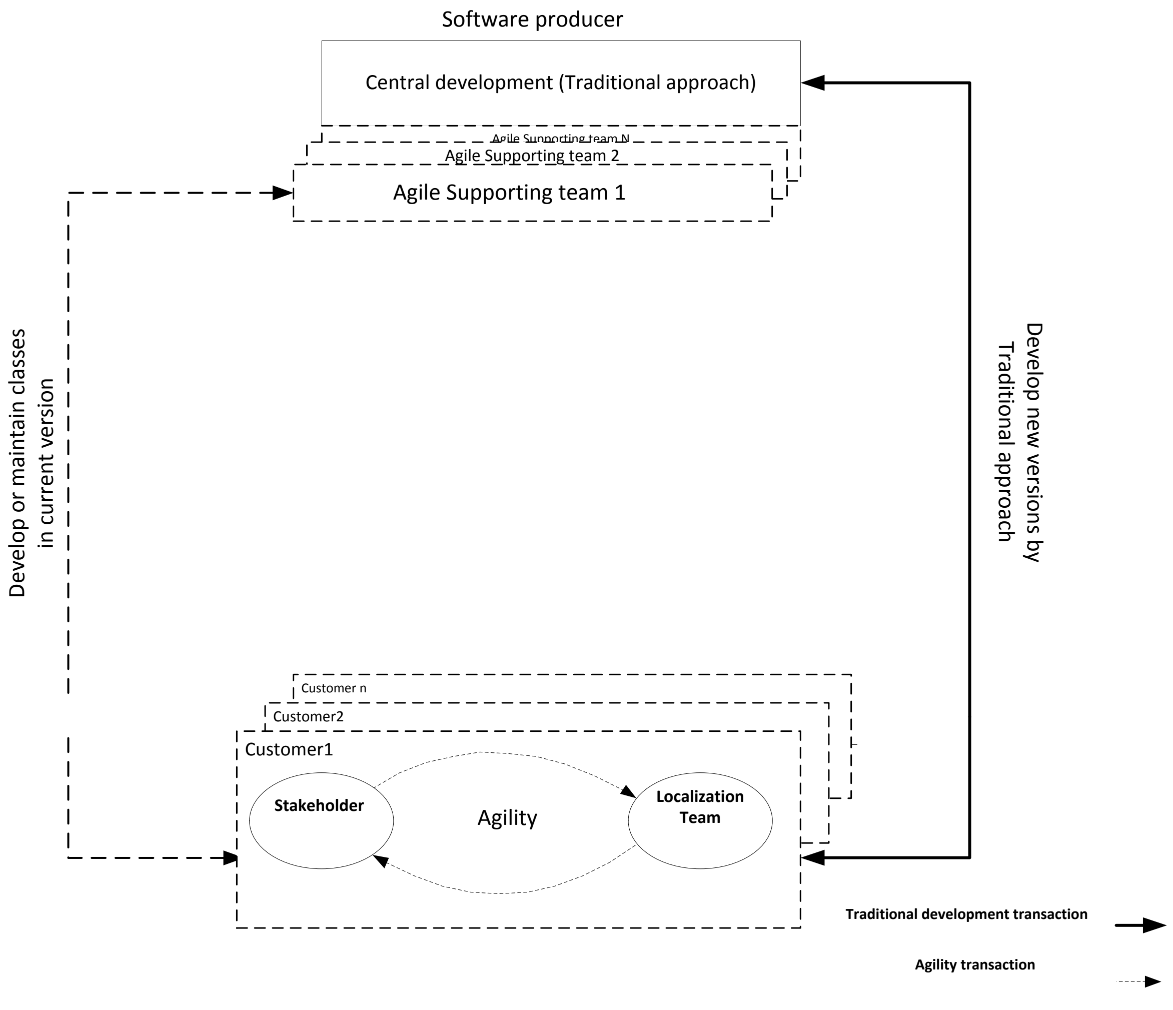


Figure 1: Agile approach across organisational boundaries.

3.3 Proposed Framework for Localisation Software across Organisational Boundaries

The proposed framework consists of four components, which are communication, project management, knowledge management, and configuration management. These components cover management aspects as well as the software development process, such as documentation and testing.

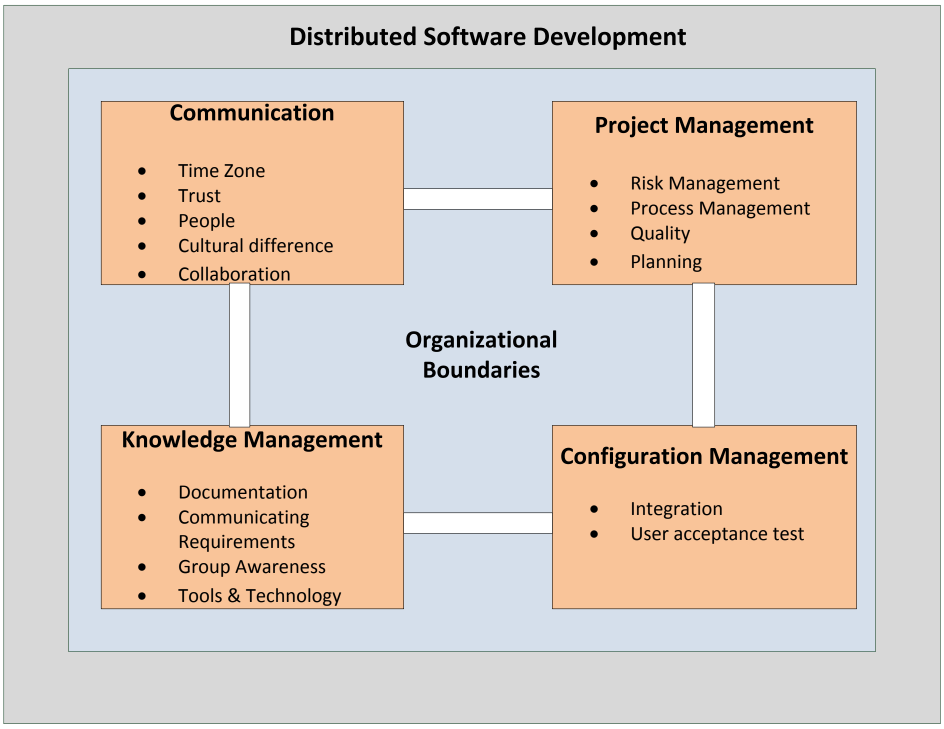


Figure 2: Framework for localisation of software product across organisational boundaries

3.3.1 Communication

Many researchers have addressed communication as one of the main issues of distributed software development as well as agile development (Fowler, 2003; Sengupta et al., 2006; Abrahamsson et al., 2002; Jiménez et al., 2009). The reason which lies behind the importance of communication is that development in general requires close communication and this requirement increases with agile development, which emphasises face-to-face communication. To discuss communication as a key factor affecting localisation software products across organisational boundaries, there are some other aspects related with communication, such as:

- **Time zone**: Time zone is an effective factor in communication, especially for teams distributed across countries as well as working hours in different organisations. Agile software development promotes people's interaction during the development process and that is difficult if there is a difference in time zone.
- **Trust:** During distributed development and development across organisations, the problem of face-to-face communication highlights another issue, which is trust between team members in different stages and forms, such as in requirements negotiation, exchange information and conveying experiences.
- **People:** The manifesto for agile software development places great emphasis on people in the development of software using agile: "*Individuals and interactions over processes and tools*" (Beck et al., 2001). Furthermore, the main motivation of the organisation in distributing their development projects is to look for highly skilled human resources (Beck et al., 2001). Structures for people in development or the localisation process across organisational boundaries, including project managers, stakeholders and developers, are a very important factor of communication.
- **Cultural difference**: This is an important factor for distributing development and developing across organisations. Fowler (2003) described cultural change as the "*hardest*" part of adopting agile methods. Also, culture can have an effect on communication, especially for global software development (GSD) projects.
- **Collaboration:** One of the four values of the agile manifesto is customer collaboration. Thus, agile software development emphasises and promotes the concept of collaboration with customers and with other developers to support that software product and the development process. In the localisation of software products across organisational boundaries, collaboration is very important to meet the customer's requirements and avoid problems of distributed sites as well as to apply the agile principles in that domain across an organisation.

3.3.2 Project management

From the literature review, project management is a hot topic for researchers in terms of applying agile principles (da Silva et al., 2010; Coram and Bohner, 2005; Lee and Yong, 2009; Hayataand Han, 2011) in distributed development, due to its effect on the development process. Those researchers have discussed different aspects of project management like this:

- **Risk management:** Risk management becomes a critical concern for people in DSD (Mudumba and Lee, 2010). In addition, these concerns increase with the application of agile principles on DSD projects or across organisational boundaries. Thus, risk management has been discussed by researchers as one of the key challenges of DSD and DAD. In the proposed framework we assumed that risk management was a part of project management and we put it as a sub component under project management.
- **Process management:** Process in the proposed framework refers to the software development process, which is clear in a traditional approach, for example in the waterfall model, analysis and design of customer's requirements implementation, testing, delivering and the documentation process. All these processes should be considered in terms of agile development and DSD across organisational boundaries. Owing to its importance in the software development process, it is addressed as a considerable component under project management.
- **Quality:** Although the software development process across organisational boundaries aims to achieve many advantages from using agile principles, like reducing the time and cost of the development process as well as increasing the productivity, the quality of produced software products take an important place. Moreover, it is addressed as an important sub-component of project management in the proposed framework.
- **Planning:** Planning takes an important place in agile development, like the planning before any iteration to sort out a priority list of the customer's requirements. However, that importance increases across boundaries to arrange the distributed development process and plans across organisational boundaries. Thus, the proposed framework gives the importance of planning in project management.

3.3.3 Knowledge management

During the development process in any software project or business, there is a huge amount of information as well as knowledge. The bulk of the information appears in different forms such as test cases, codes, comments and logs on source codes, project specifications and developers' and project team members' experiences and comments. Furthermore, this information should have a level of accuracy and availability through useful tools. The proposed framework emphasises knowledge management and integration as key components in software development across distributed multi-teams.

- **Documentation:** The manifesto for agile software development puts the emphasis on working software over comprehensive documentation. However, documentation in DSD and across organisations is required to solve the lack of face-to-face as well as informal communication. Herbsleb and Moitra (2001) discussed documentation in GSD and they emphasised the documentation process in DSD as part of the knowledge management.
- **Communicating requirement:** The proposed framework promotes management practice and software engineering practices through agile concepts. Agile software development support face-to-face communication and interaction with customers over the complexity of the process. The framework supports the idea of allocating agile teams in the customer's location to gather customer's requirements and other agile teams in distributed development to deal with these requirements.
- **Group awareness:** Information should be available as well as equal to the people in distributed agile development teams, like developers in different sites. Thus, group awareness is a very important factor. Hence one of the manifesto's values is an emphasis on individuals and interactive action.
- **Tools and technologies:** In the development process, either using traditional approaches or agile methodology, some tools and technologies are used. Those tools can be at the communication level or at the development and management level, like tracking tools and documentation tools.

3.3.4 Configuration and integration management

The coordination and synchronisation of the source code and software versions is an important step for any iteration development. However, the integration and version control of the source code becomes more complex with distributed projects across multi-teams and organisations. Therefore, configuration management is a key component in the proposed framework and it guides the developers and project managers at the customer's location so that they consider this step and make sure the new version of any iteration is integrated with the customer's needs and the customer's environment in terms of both platforms and hardware.

- **Integration:** For the localisation process of software products across organisational boundaries, there are multi versions to meet customers' change requests or new requirements. Thus, the integration process is emphasised to make sure the new version is compatible with the current version to or customise that version for the organisation system. Also, emphasis is put on using the version control concept and technology to work as well as move smoothly from version to version in the localisation process across customers' boundaries.
- **User acceptance test**: Most software testing happens in development time by the development team, like unit tests and integration tests. However, user acceptance tests require sharing customers in this kind of test to make sure that the software meets all customers' requirements. Thus, the user acceptance test is the one of key components of the proposed framework to address the testing process across organisational boundaries.

3.4 Summary and discussion of proposed framework chapter

The research objectives were to introduce the agile concept and propose a framework to address key factors of that system localised across organisational boundaries. In this chapter, we discussed how agile principles could be applied in distributed development projects across organisational boundaries to localise software products in terms of applying the customer's requirements and requests in a current version of that product. In addition, we discussed how this adoption of agile principles would support project managers, developers and stockholders.

Furthermore, we proposed a framework to address key factors of management and software engineering aspects for the localisation process across organisational boundaries. The introduction of and proposed framework for agile options might decrease the challenges of DSD and development across boundaries.

Chapter 4 . Conclusion and Future work

4.1 Conclusion

As we discussed previously in this report, distributed software development is the new trend in software development, as well as agile software development being a departure from the traditional approaches, like the waterfall model. Furthermore, in the reviewing of distributed development across organisational boundaries, we identified the research gap, which was the lack of a suitable framework for management and software engineering aspects for the localisation of software products across organisational boundaries. Furthermore, there are many examples to support that motivation to investigate localisation software products across organisational boundaries. The example from literature is the My Yahoo! 'Chameleon' project, which aims to localise web software products in international locations based on the agile process (Lee & Yong, 2009). In addition, from my own experience, I have worked for three years to represent my employer, fronting a localisation team to develop and localise administration software products with distributed support, the same scenario as shown in Figure 1 (Taif University, 2008).

Our idea was to fill in that gap by introducing agile software development to localisation projects across organisational boundaries and proposing a framework to address the key factors of the localisation process using agile principles.

Our goals in this research are:

I. Introducing agile software development to the localisation process for software products across organisational boundaries.
II. Proposing a framework based on agile principles. The proposed framework would have a combination of management aspects like project management and software engineering aspects such as communicating requirement, documentation and testing. In addition, it may support people such as project managers, developers and stakeholders to understand the organisational domain and the key factors and challenges in that domain.

4.2 Research Questions

Q1 - How can we introduce agile software development principles to localisation software products across organisational boundaries?

Q2 – How would agile software development improve / help in the localisation process for software products across organisational boundaries?

Q3 – How would the proposed framework help / support people in the localisation process across organisational boundaries?

4.3 Future work

This research aims to introduce agile development principles to the localisation of software products across organisational boundaries, and also to propose a framework for this domain to address the key factors of using agile software development. The future work, after the stage that has been presented in this report, will be divided into four steps (Figure 4):

- **Review the proposed framework**: In this step, we will check the design of that proposed framework. We could use the triangulation concept (using three ways to prove the result) to prove and improve the framework by reviewing the literature, find a case study from the literature to compare the input as well as the output of the proposed framework and review that framework and introduce agile into organisational boundaries with the most agile practitioners to get their feedback. Figure 3 shows the Gantt chart of the plan and its milestones (Figure 3).

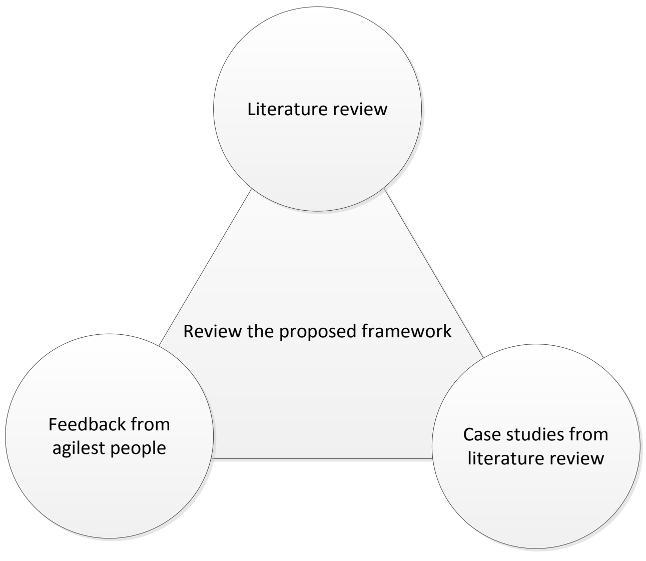

Literature review
Review the proposed framework
Feedback from agilest people
Case studies from literature review

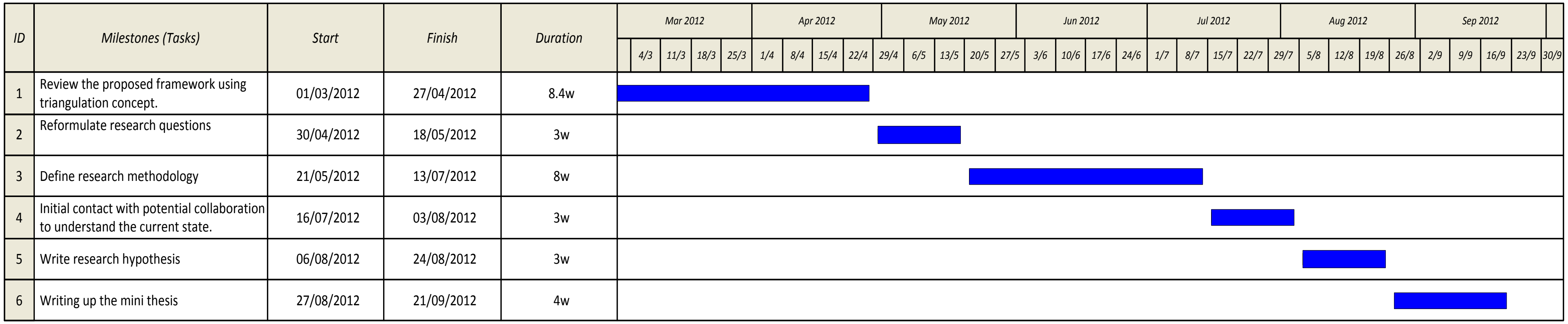
Figure 3: Triangulation review of proposed framework

- **Formulate the research questions**

Research questions should lead to the research goal and state what the research will investigate. In this stage we are going to discuss what kind of questions we need in this research, and then formulate appropriate questions for this research.

- **Choose and define the research methodology**

In terms of research methodology, it is an important step to identify which are the appropriate research methods to follow. Through the selected methodology, we could prove and evaluate the research results. This step will take place after formulating the research questions.



ID	*Milestones (Tasks)*	*Start*	*Finish*	*Duration*
1	Review the proposed framework using triangulation concept.	01/03/2012	27/04/2012	8.4w
2	Reformulate research questions	30/04/2012	18/05/2012	3w
3	Define research methodology	21/05/2012	13/07/2012	8w
4	Initial contact with potential collaboration to understand the current state.	16/07/2012	03/08/2012	3w
5	Write research hypothesis	06/08/2012	24/08/2012	3w
6	Writing up the mini thesis	27/08/2012	21/09/2012	4w

Figure 3: Gantt chart of future plan milestones

References

Abrahamsson, P., Salo, O., Ronkainen, J., & Warsta, J. (2002). Agile software development methods: review and analysis. *VTT Technical report*.

Akbar, R., Haris, M., & Naeem, M. (2008). Agile Framework for Globally Distributed Development Environment (The DAD Model). 8th WSEAS International Conference on Applied Informatics and Communications Rhodes, Greece, pp. 423-428.

Beck, K. (1999). *extreme Programming Explained: Embrace Change*. 2nd ed . *XP Series* (p. 224). Addison-Wesley Professional

Cohen, D., Lindvall, M., & Costa, P. (2004). An Introduction to Agile Methods. *Advances in Computers*, *62*(03), 1-66.

Coram, M., & Bohner, S. (2005). The Impact of Agile Methods on Software Project Management. *Proceedings of the 12th IEEE International Conference and Workshops on the Engineering of Computer-Based Systems (ECBS'05)*.

Damian, D., & Zowghi, D. (2003). Requirements Engineering challenges in multi-site software development organizations. *Requirements Engineering Journal, 8*, pp.149-160.

Dingsøyr, T., Dybå, T., & Abrahamsson, P. (2008). A Preliminary Roadmap for Empirical Research on Agile Software Development. *Agile 2008 Conference*, pp. 83-94. IEEE.

Fowler, M. (2001). The new methodology. *Wuhan University Journal of Natural Sciences, 6*(1-2), 12-24.

Fowler, M. (2003). Using an Agile Software Process with Offshore Development. *Development*, 1-8.

Fowler, M., & Highsmith, J. (2001a). *The Agile Manifesto*. *Software Development* (Vol. 9, pp. 28–35). [San Francisco, CA: Miller Freeman, Inc., 1993-. Retrieved from http://andrey.hristov.com/fht-stuttgart/The_Agile_Manifesto_SDMagazine.pdf

Glass, R. L. (2001). Agile versus traditionlal: Make love, not war. *Cutter IT Journal*, *14*(12). Retrieved from http://www.cutter.com/content/itjournal/fulltext/2001/12/itj0112c.html

Hayata, Tomohiro and Han, J. (2011). A Hybrid Model for IT Project with Scrum. *Work*, 285-290.

Hossain, E., Babar, M. A., Paik, H.-young, & Verner, J. (2009). Risk Identification and Mitigation Processes for Using Scrum in Global Software Development: A Conceptual Framework. *2009 16th Asia-Pacific Software Engineering Conference*, 457-464. IEEE.

Jiménez, M., Piattini, M., & Vizcaíno, A. (2009). Challenges and Improvements in Distributed Software Development: A Systematic Review. *Advances in Software Engineering*, *2009*, pp.1-14.

Kent Beck, Beedle, M., Bennekum, A. van, Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., et al. (2001). Manifesto for Agile Software Development. Retrieved February 14, 2012, from http://agilemanifesto.org/

Lee, S., & Yong, H.-S. (2009). Distributed agile: project management in a global environment. *Empirical Software Engineering*, *15*(2), pp. 204-217.

Moitra, J. D. H. and D. (2001). Global software development - IEEE Software. *IEEE Software*, (April), pp. 16-20.

Morien, R., & Wongthongtham, P. (2008). Supporting agility in software development projects - defining a project ontology. *2008 2nd IEEE International Conference on Digital Ecosystems and Technologies*, pp.229-234. IEEE.

Mudumba, V., & Lee, O.-K. (Daniel). (2010). A New Perspective on GDSD Risk Management: Agile Risk Management. *2010 5th IEEE International Conference on Global Software Engineering*, pp. 219-227. IEEE.

Phalnikar, R., Deshpande, V. S., & Joshi, S. D. (2009). Applying Agile Principles for Distributed Software Development. *2009 International Conference on Advanced Computer Control*, pp. 535-539. IEEE.

Rodríguez, J., Ebert, C., & Vizcaino, and A. (2010). Technologies and Tools for Distributed Teams. *IEEE Software*, *27*(5), pp.10-14.

Santos, F. M., & Eisenhardt, K. M. (2005). Organizational Boundaries and Theories of Organization. *Organization Science*, *16*(5), pp. 491-508.

Sengupta, B., Chandra, S., & Sinha, V. (2006). A research agenda for distributed software development. *Proceeding of the 28th international conference on Software engineering - ICSE '06*, 731. New York, New York, USA: ACM Press.

Shore, J., & Warden, S. (2007). *The art of agile development* (First.). O'Reilly.

Sureshchandra, K., & Shrinivasavadhani, J. (2008). Adopting Agile in Distributed Development. *2008 IEEE International Conference on Global Software Engineering*, 217-221. IEEE.

Taif University. (2008). Internal business documents private communications.

Wahyudin, D., Heindl, M., Eckhard, B., Schatten, A., & Biffl, S. (2008). as Formal Means to Balance Agile Practices in Global Software Development Settings. *IFIP International Federation For Information Processing*, pp. 208-222.

da Silva, F. Q. B., Costa, C., Franca, a. C. C., & Prikladinicki, R. (2010). Challenges and Solutions in Distributed Software Development Project Management: A Systematic Literature Review. *2010 5th IEEE International Conference on Global Software Engineering*, 87-96. IEEE.

Šmite, D., & Borzovs, J. (2006). A Framework for Overcoming Supplier Related Threats in Global Projects. Software Process Improvement. *Proceeding, Lecture Notes in Computer Science, vol. 4257, Springer-Verlag, 2006, pp. 50-61.*