UNIVERSITY OF GOTHENBURG

Improving Global Software Development Tools to Facilitate Distributed Agile Development

Bachelor of Science Thesis in Software Engineering and Management

Zarif Jawad

Makiko Taira

University of Gothenburg
Chalmers University of Technology
Department of Computer Science and Engineering
Gothenburg, Sweden 2013

Improving Global Software Development Tools to Facilitate Distributed Agile Development

Zarif Jawad

Makiko Taira

Examiner: Lars Pareto

University of Gothenburg
Chalmers University of Technology
Department of Computer Science and Engineering
SE-412 96 Göteborg
Sweden
Telephone: +46 (0)31-772 1000

Department of Computer Science and Engineering
Göteborg, Sweden 2013

Improving Global Software Development Tools to Facilitate Distributed Agile Development

Zarif Jawad
gusjawza@student.gu.se

Makiko Taira
gustairma@student.gu.se

IT University of Gothenburg, Software Engineering and Management, Gothenburg, Sweden

CHALMERS GÖTEBORGS UNIVERSITET

Abstract

❋ ❋ ❋ ❋ ❋ ❋ ❋ ❋ ❋ ❋ ❋ ❋

Using Agile processes within distributed teams is becoming common in many software development environments. Yet, it is certain that the globally distributed teams will result in problems as well as benefits. This paper attempts to discern how GSD (Global Software Development) tools can be improved to better support distributed Agile processes by implementing a virtual task board prototype. From our research, we could draw the conclusion that virtual task boards are crucial feature for task management GSD tools and additional advantages can be achieved by implementing this feature on top of an existing collaboration and document management platform.

Keywords: *Agile, Software, Development, Kanban, Distributed*

1 Introduction

A trend during recent decades in the software development industry is the adoption of Agile Software Development (ASD) methods (Aitken & Ilango, 2013). Several early implementations of Agile development had been focusing primary on the development process in software projects. In recent years, the attention has moved to issues related to actual management of Agile projects, e.g. Agile planning, control and estimation, and streamlining flow of stories (Dingsøyr, et al., 2012). The common characteristics of ASD methods include self-organizing teams, iterative development and the ability to efficiently respond to requirement changes during a project (Dingsøyr, et al., 2012) (Aitken & Ilango, 2013).

Agile provides many benefits: transparent project progress, better control of projects, early detection of project issues, reduced overall risks associated with software development, increased customer satisfaction (Kajko-Mattsson, et al., 2009).

One way that Agile development teams organize themselves is by using task boards. Task boards are physical or digital boards that represent tasks and their respective statuses; in other words, they are used to visualize the work (Hajratwala, 2012). Task boards are traditionally created using notice boards or sticky notes. Each note is called a “story card” and represents a user story or work item. The board has vertical columns representing stages in the development process (see Figure 1). The notes are moved between the columns on the board as the work item or user story progresses in the process. Information about whom the task is assigned to, when it was started and when it was completed is added to the story card as appropriate. This information can be used to calculate different projects metrics such as velocity, lead times and bottlenecks.

Task boards works well in a co-located environment, by making the collaborative environments stronger. However, using a physical task board may cause several problems when Agile teams are distributed. Wang, et al. (2010)

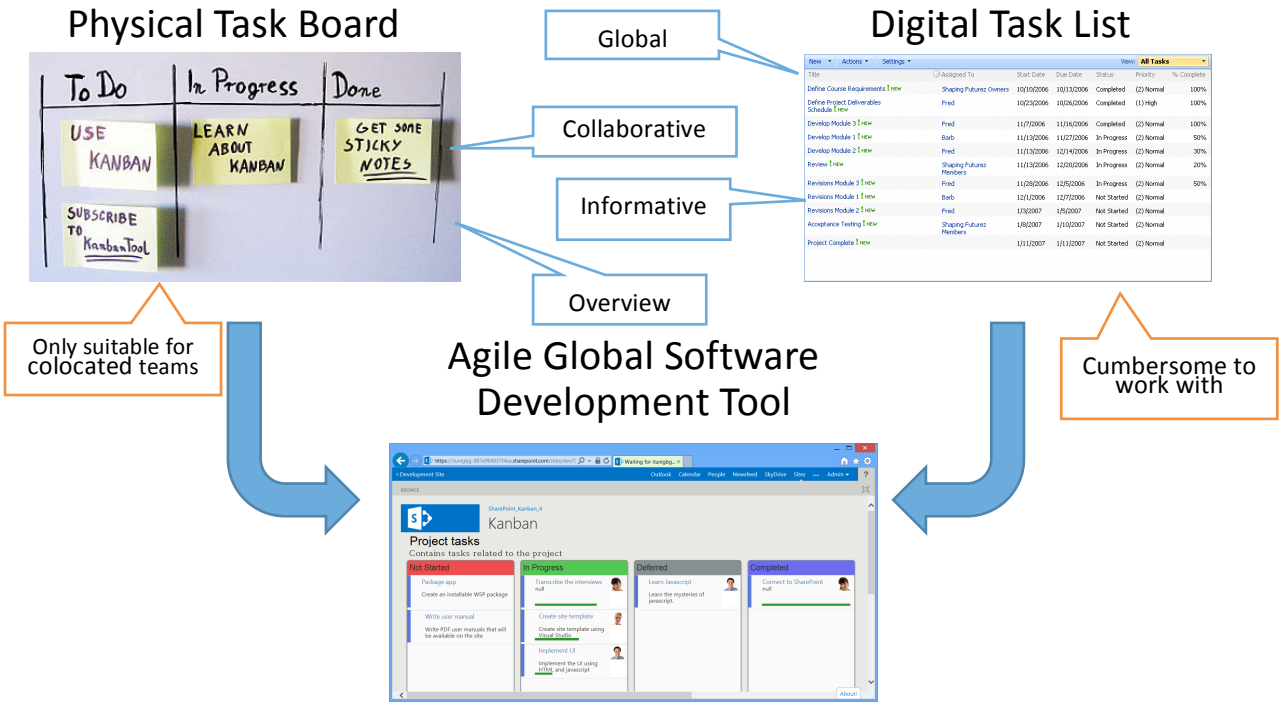


Figure 1: Task Management Evolution

shows that the risk of misunderstandings increase since distributed teams replicate story cards to each site. Also, obviously, since there is less communications in distributed teams than in co-located ones, there is a high possibility that problems will not be reported until they become significant enough (ibid). Teams might misunderstand each other or be unaware of the way another team works (Liskin & Schneider, 2012).

A trend happening in parallel with ASD is Global Software Development (GSD): "software work undertaken at geographically separated locations across national boundaries in a coordinated fashion involving real time or asynchronous interaction" (Sahay, et al., 2003). Today many companies develop software in distributed teams across many business worksites, locations and countries (Phalnikar, et al., 2009). GSD reduces costs and in some cases, the time to market by using 24-hour development (ibid).

Although distributed development is an integral part of GSD, many GSD tools available on the market today do not provide enough support to facilitate distributed Agile development.

This led us to the following research problem:

GSD tools must be improved to better support collaboration and communication in distributed Agile development projects.

To examine the Research Problem the study was carried in collaboration with a company practicing agile methodologies and working in a distributed environment with particular interest in realizing virtual task boards using SharePoint.

Already at an early phase of the study, it was discovered that the company was using a GSD tool for task management that provided the virtual Kanban interface. At the same time, they had a separate system for collaboration and document management. This system also provided features for task management, however, this part of a system was never used since it was considered too primitive to be productive by limitations that turned out to have major implications of ASD practices.

In response to this, we decided to focus on providing a virtual task board user interface on top of the existing collaboration and document management system.
Our research question, accordingly, is as follows:

What features are needed for virtual task boards to be able to utilize them in distributed ASD processes?

In order to answer our research question, we developed a virtual task board user interface as an add-on to an existing task management system under the assumption that it would go along way towards aligning GSD and ASD methods. Also, a literature review and interviews were carried out

to understand which features are most crucial for a virtual task board and to gauge the usefulness of the provided prototype. To validate our results, presumptive users were given a demonstration of the tool after which they evaluated it during the following interviews.

Through this research, it was discovered that virtual task boards improve the communication between geographically distributed team members, reduce the risk of project delays and help the entire team to follow the overall progress of the project in a distributed Agile environment.

This paper is organized into the following sections: The first section introduces the reader to distributed Agile development and the aim of this paper, the second section explains the research method, the third and fourth sections presents the result, the fifth and sixth section details the requirements for the prototype that was developed. Section seven contains information about how the prototype was implemented. Sections eighth and ninth present the results we could observe using the prototype and our findings based on the analysis of our results. The tenth and final section contains the conclusion we could draw from our research.

2 Methodology

In the field of Agile, Design Research is a commonly practiced research method to analyze, formulate, and improve engineering challenges (Henver, et al., 2004). We chose Design Research as the Research method for our study because the scope of our research covered the improvement of Agile processes in a Distributed environment which could be categorized as an SPI activity (McFeely, 1996).

2.1 Research Setting

The study was conducted at Mogul Gothenburg AB. The organization is a consultant company with offices in Stockholm, Gothenburg, and Belgrade, practicing Agile methodologies particularly Scrum and Kanban. Mogul Gothenburg AB expressed their interest in learning the different challenges involved in distributed Agile development and solve some of these issues.

One of the main interests of the company is the Microsoft SharePoint platform because the company currently has a lot of projects and customers using SharePoint. Accordingly a prioritized requirement for Mogul was that the prototype be implemented using the SharePoint platform. Therefore it was decided to implement the prototype in SharePoint 2013.

SharePoint is a completely web based platform for ECM (Enterprise Content Management) and WCM (Web Content Management) from Microsoft first launched in 2001. It is both a platform and a toolbox for extending that platform and has many applications such as corporate intranets, extranets and publicly facing web sites. It has a wide range of features, for example: document management, collaboration and business intelligence. This made SharePoint a good platform on which to build the virtual task board prototype.

2.2 Research process

Design Research was chosen as our method of study, as our study was part of "Product design" due to the implementation of a prototype and followed the real research category (Vaishnavi & Kuechler, 2004).

To carry out our study following a design research pattern we went through the process of achieving awareness of the problem, abduction and deduction, as prescribed by Vaishnavi & Kuechler (2004). Subsequently the following steps were conducted: (i) Literature Review, (ii) Interviews, (iii) Existing tool evaluation, (iv) Requirements and Design, (v) Prototype Development, (vi) Prototype Evaluation, and (vii) Conclusion (see Figure 2).

Following the aforementioned process, we implemented a virtual task board as a tool that would be accessible by teams across short and long distances and help coordinate multiple teams.

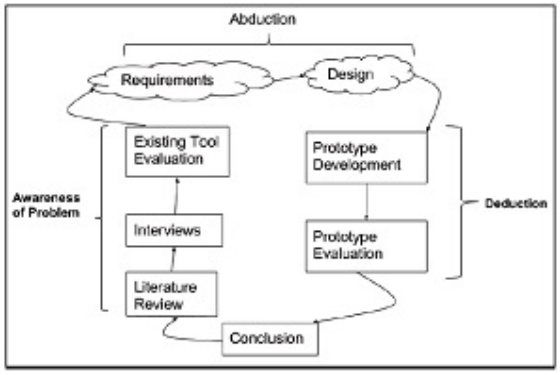


Figure 2: Reasoning in the Design Cycle (Vaishnavi, 2004)

2.2.1 Literature Review

To discover the challenges in distributed ASD we began by carrying out literature review. Firstly literature review journals and articles were searched for and reviewed as these are concrete sources of information on the rapidly changing GSD field (Booth, et al, 2003). Keywords were used in order to search for the research papers, which are listed below:

- Global Software Development
- Distributed Agile Software Development
- Challenges in Global Software Development
- Challenges in Distributed Agile Software Development

In total 17 different articles were found related to our study. The papers were then scrutinized to come up with the best research papers to conduct our literature review. This was done skimming through abstracts, introductions and conclusions. In total 7 different research papers were identified to be the most suitable literature for our study. The selected papers were then systematically reviewed in order to gain knowledge about distributed ASD and its challenges. Lastly, thematic analysis was used to categorize these findings to come with results of this step.

2.2.2 Interviews

Since the focus of our study was addressing distributed ASD challenges, we conducted interviews of 6 ASD practitioners at Mogul working at different geographic locations in Serbia and Sweden. We interviewed both project managers and developers to get a better view of the distributed ASD practices and problems in the company. The interview recordings were then transcribed and analyzed to come up with the findings. The findings were in turn categorized using thematic analysis to produce the results of Interviews.

The data collected through both interview and literature review were then scrutinized to understand the problems and challenges with Agile processes in GSD. The results from the analysis were then used to come up with requirements for our prototype and also recommendations on how Agile processes can be improved by using GSD tools.

2.2.3 Existing Tool Evaluation

To raise our awareness of the research problem, existing GSD tools were also evaluated. Two specific tools were selected, by ranking the tools that were found to be the most popular in the interview analysis. This was done by first listing out all the GSD tools mentioned by the interviewees and then ranking them based on the number of times each were mentioned in the interviews. The tools that were selected are JIRA and SharePoint task list. Evaluation of tools was done by skimming through instructions and extensively using the tools. By doing so it helped us to understand the problems with existing tools used by Mogul, and also assisted in formulating the requirements of our prototype.

2.2.4 Requirements and Design

Once the steps (i), (ii) and (iii) were completed we had the necessary information to come up with the requirements for our prototype. The requirements specified would allow the design and implementation of a prototype, which would address distributed ASD challenges by providing full utilization of Task boards.

With the requirements specification completed we moved on by designing the architecture for our prototype, to begin the implementation of the prototype.

2.2.5 Prototype Development

With architecture for the prototype, we began the implementation of our virtual task board. The goal of our effort was not to develop a full-featured version of the virtual task board but rather implement the most relevant feature possible within the timeframe. We set a deadline for the completion of our prototype and divided our time up into sprints and set the tasks for each individual sprints. At the end of each sprint, we held a sprint review meeting with our supervisor to demonstrate the progress made in each sprint.

2.2.6 Prototype Evaluation

To ensure that the results from the user evaluation of the virtual task board prototype would be related to it's general feature set and usability, a series of manual system tests were performed before the prototype could be considered ready for evaluation. However, since it was in fact a prototype, the goal was not to achieve zero defects but merely to avoid negative feedback as a result of programming mistakes.

Since SharePoint task lists were more extensively used by Mogul's customers, one of which was a European car company with over 20,000 SharePoint users, the first stable version of the prototype was evaluated by two Agile

practitioners from this company using SharePoint task lists in their daily work. They were given a demonstration of the prototype, which was followed by an interview on their opinion about the prototype (see Appendix B). Using the questions, we tried to find out whether or not the prototype and the several other possible features together would help address the issues in distributed ASD.

2.2.7 Conclusion

In this step we tried to map and find out whether the findings with our research in combination with our prototype and possible features were solutions addressing our research question.

3 Result: Challenges recognized through Literature review

To conduct our literature review we searched for journals and articles using IEEE and several other journal databases. We found 17 different research papers with relevant topic to our thesis title. The articles were then sorted to find out the most prominent challenges in distributed agile development. Table 1 shows an overview in the findings of Literature review.

Traditionally Agile planning and development is conducted in a co-located environment (Wang, et al., 2010). In the following sections we describe the challenges that arise when using Agile planning in a distributed environment.

3.1 Task Management

In software development organizations practicing Agile methodologies, projects are usually planned before starting with development. The project time is usually divided up into sprints and the tasks are attached to each sprint based on their priority (Guang-yong, 2011). The projects are then assigned to development teams and the tasks are divided up among the developers within the teams based on their skills and experiences (ibid). After assigning the tasks it becomes crucial for all the stakeholders of the project, including project managers, developers, customers, etc. to track the progress of the tasks (Hajratwala, 2012). This process of assigning and tracking tasks are vital to the effective progress and functioning of a software development project. Project managers and use physical task boards to help visualize the process of task management and keep track of it (ibid). But this process of task management becomes even more complex when the process is carried out in a geographically distributed environment. The stakeholders have to rely on virtual task boards in order to carry out the process of task management in a virtual environment. But the process of task management comes with a lot of challenges involved such as the following.

- Time waste on explaining detailed functionality (Hajratwala, 2012).
- The physical distance making it harder to communicate regarding task management or, project issues (Herbsleb & Moitra, 2001).
- In-appropriate work distribution among developers with respect to experience and skills (Hajratwala, 2012).
- Dissatisfaction among developers over task and time assignment (Herbsleb & Moitra, 2001).

3.2 Project Delays

Software Development projects are usually faced with deadlines but in order to meet the deadlines the time required to complete each tasks are planned in advance (Guang-yong, 2011). But due to certain issues such as inappropriate time estimates, integration delays or bug fixing, etc., it is common

to miss deadlines (Herbsleb & Moitra, 2001). In the case of distributed development teams this might lead to even bigger problems as tasks are quite often divided among distributed locations and teams wait for each other to complete their tasks (ibid). In case any team at a certain location misses their deadline

to submit their code or complete their task it might lead to delays to other tasks that depend on it.

The problem of project delays are caused due to some of the following reasons:

- Defective project Planning (Liskin & Schneider, 2012).
- Inadequate communication between project managers and developers (Wang, et al., 2010).
- Synchronous Agile planning meetings not carried out properly (Wang, et al., 2010).

Table 1: Overview of Literature Review

Challenges	Task Management	Project Delays	Team Issues	Communication
Reasons	Time Waste on explaining detailed functionality (Hajratwala, 2012)	Defective Project Planning (Liskin & Schneider, 2012)	Unbalanced teams (Liskin & Schneider, 2012)	Ignoring small problems (Herbsleb & Moitra, 2001)
	Physical Distance (Herbsleb & Moitra, 2001)	In-adequate communication between stakeholders (Wang, et al., 2010)	Team morale (Hajratwala, 2012)	Time-zone differences (Herbsleb & Moitra, 2001)
	In-appropriate work distribution (Hajratwala, 2012)	Improper execution of meetings (Wang, et al., 2010)	Cultural Issues (Herbsleb & Moitra, 2001)	Decreased visibility on Project status (Phalnikar, et al., 2009)
	Dis-satisfaction among developers (Herbsleb & Moitra, 2001)		Lack of experts at distributed sites (Liskin & Schneider, 2012)	Language Issues (Herbsleb & Moitra, 2001)
				Data exchange between different Agile tools (Herbsleb & Moitra, 2001)

3.3 Team Issues

Software development is usually carried out by a team of people having different designations and roles (Guang-yong, 2011). In case of Agile, the team usually consists of a Scrum master and Product owner followed by a team of developers (ibid). The work process followed by these teams are usually one of the Agile processes such as Scrum, Kanban, XP, Lean, etc. But during the

development process a lot of issues might arise within the teams as the teams are usually not formed by themselves but rather the by managers. Distributing the project work among several different geographic locations creates even more issues among the different stakeholders. Some of these challenges were mentioned in several literature reviewed:

- Unbalanced teams such as lack of experts. (Liskin & Schneider, 2012)
- Team morale and energy very low (Hajratwala, 2012)
- Cultural difference of team members creates communication issues. (Herbsleb & Moitra, 2001)
- Lack of Scrum master and Project manager at both locations. (Liskin & Schneider, 2012)

3.4 Communication

One of the key challenges in ASD is communication, as ASD requires constant communication between the different stakeholders in the project (Kamaruddin, et al., 2012). In a collocated environment it might not be such an important issue, but in a distributed environment, communication between the different stakeholders becomes a challenge due to the physical distance between teams (ibid).

Many companies try to avoid these issues by allowing travelling between the distributed locations, but due to travel complications such as visa issues and also this being an expensive undertaking it might not be the best way to address this challenge (Battin, et al., 2001).

Some of the Key challenges in a distributed environment are:

- Problems not being reported until considerably large making it harder to resolve (Herbsleb & Moitra, 2001).
- Time-zone differences make it harder to decide on meeting time (Herbsleb & Moitra, 2001).
- Decreased visibility on Project status or progress (Phalnikar, et al., 2009).

- Multiple languages not being supported in many GSD tools (Herbsleb & Moitra, 2001).
- Data exchange between different Agile planning tools are problematic (Herbsleb & Moitra, 2001).

4 Interview Result: Challenges recognized by practitioners

The interviews consisted of 4 face-to-face interviews and 2 Skype interviews (Appendix A). The data collected from these interviews are based on the responses to the questions asked and then divided into respective sections: background information, challenges, and solution tools for communication problems.

Figure 3 shows the distribution of the challenges that the interviewees perceive in using distributed Agile development processes. The result of the interview as visualized shows that an overwhelming majority of interviewees think that language and psychological barriers are the main problems.

Both of these problems are related to communication. Together with other related challenges, communication problems can be divided into 3 categories: *language*, *customer-facing* and *psychological distance barriers*.

4.1 Language

Most interviewees think that language can be an issue between the two teams even though all of them are fluent in English. Communicating with one another in their mother language is much easier and speaking in their second language is sometimes prone to misunderstanding.

"We are Swedish and we need to speak English and those in Belgrade need to speak English. Since none of our first languages are English, there is also a [risk of] misunderstanding and sometimes it could possibly also be hard to inform or give the whole picture of an issue or project" said one of the project managers.

Language is not only a problem during conversations. Since they get email in their native language, it is impossible to just forward them to the offshore team, but there is no time to translate all email which makes it difficult for them to understand the whole picture of the topic being discussed.

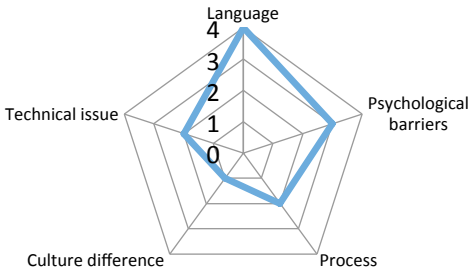


Figure 3: Radar chart of perceived challenges

As another project manager put it:

"It's much easier if you can speak your own language [...] much mail conversations are in Swedish that is of course a problem. Because I can read mail in Swedish and understand it right away. You don't translate the mail and send it to people from Belgrade. You cannot do that with all mail, so more difficult to get them, really getting a picture on the overall process of how things works"

There are of course some automated translation tools. However, they cannot solve the problem completely.

During the interviews, one project manager stated:

"They use Google translate or some automated translation service which is also a risk because it's not a 100% guarantee that they understand correctly the Google translate understand the Swedish correctly"

Additionally, since a customer-centered design is one of the main features in Agile methods, it is often obligatory to communicate in the native language of the customer.

"For most developers in Gothenburg, English is not a big problem [...] But the problem is that in the Agile processes, the customers are supposed to involve in the processes and they don't always feel comfortable communicating in English. [Customer thinks:] We are Swedish company you are Swedish company, too and why do we have to talk in English for this project. Doesn't make really sense", said one of the interviewed developers.

4.2 Process

There are several challenges related to the software development process itself. It is difficult to perform code reviews with offsite teams. Also it is very important to communicate with each other about social matters for instance, outside of meetings, e.g. asking, "How are you?" or "How was your weekend?" to strengthen the team spirit. Additionally, Agile methods often include customer-involvement, and this is difficult for offshore teams that cannot communicate directly with customers. The two interviewees who work as project managers both state that the inability to participate in face-to-face meetings is one of the problems. Getting information through a task board is not enough since it is difficult to get a sense of the whole picture of what the customer really want.

As they put it themselves:

"Even if they sometimes are in meetings with customers, they only hear what's said on that meeting and they know what's written in JIRA but they don't hear they talk between us and between the customers. Here we go to customers, we talk to the customers, go to customers then we understand so much between the lines. That's actually one of the biggest problem, communication to get them understand exactly the picture on what the customer wants"

And:

"I am sitting with customer so I have very good at what we supposed to do but they are on the other side of Europe so just putting of this technical terms often doesn't turn out well, so very important to sit down what is the customer supposed to do with this features"

However, developers in the offsite team seem to be less concerned with the effect that distance can have on communication between the teams.

"If we talk about the communication it also can be some kind of problem but so far we have Skype, Lync, Tandberg video conference and so on. So even that is not the problem anymore" said one of the developers in the offsite team.

4.3 Psychological distance barriers

The increased effort required to contact a team member in a different location create a psychological barrier that doesn't exist between people sitting in the same room. This means that team members may refrain from communicating until the perceived need exceeds this barrier.

"If you sit together, you have the barrier to communicate to other people is smaller. If there is something strange I can just turn around and tap his shoulder […] just very quickly gives feedback and get some help […] But our colleagues in Belgrade, they don't do that. They contact me for something they say excuse me, do you have time? I would like to talk to you about something. Yeah, sure. Just wait second. I have to move to another room, so they take their computer and they go to other room and they start Skype because they don't want to disturb other people" said one of the architects.

A developer in the offsite team has the exact same problem:

"you start by chat if you hear me something to talk to somebody, if video is not available, move to chat, so it should be ever more than like a minutes of holding you know"

4.4 Culture Differences

Culture and social class differences can sometimes cause problems, so it is important that they try to understand each other.

"There is also social class differences, you see that there is actually the first and second classes […] some people are paid better or are more educated and stuff like that. This is the social aspect all cross the world", one project manager said.

4.5 Technical Issue

Having problem with internet connectivity or some GSD tools may interrupt the communication between teams as one of the offsite team members pointed out:

"There are always infrastructural problems, for example, you need to provide good internet connection that should be very reliable and very fast"

5 Requirements Specification through tool evaluation

To come up with requirements for our prototype we evaluated two current GSD tool used by Mogul and its customer. The evaluation was carried out by extensive usage of the tools. We

were granted access into some existing JIRA projects at Mogul (see Figure 4).

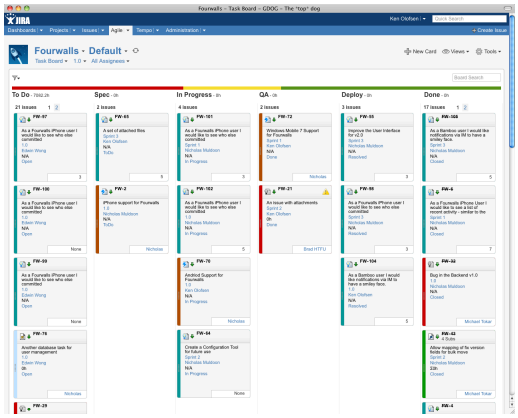
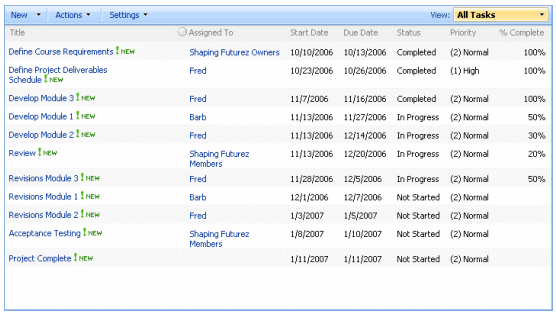


Figure 4: JIRA

Incase of SharePoint task lists, a task list was created on Microsoft Office365 developer account and then populated with fake tasks (see Figure 5). The functionalities provided by the tools were noted and considered as a standard requirement. We then combined these requirements with several features recognized through literature review and interviews in order to come up with a full set of requirements (see Table 2).



Title	Assigned To	Start Date	Due Date	Status	Priority	% Complete
Define Course Requirements	Shaping Futurez Owners	10/10/2006	10/13/2006	Completed	(2) Normal	100%
Define Project Deliverables Schedule	Fred	10/23/2006	10/26/2006	Completed	(1) High	100%
Develop Module 3	Fred	11/7/2006	11/16/2006	Completed	(2) Normal	100%
Develop Module 1	Barb	11/13/2006	11/27/2006	In Progress	(2) Normal	50%
Develop Module 2	Fred	11/13/2006	12/14/2006	In Progress	(2) Normal	30%
Review	Shaping Futurez Members	11/13/2006	12/20/2006	In Progress	(2) Normal	20%
Revisions Module 3	Fred	11/28/2006	12/5/2006	In Progress	(2) Normal	50%
Revisions Module 1	Barb	12/1/2006	12/7/2006	Not Started	(2) Normal	
Revisions Module 2	Fred	1/3/2007	1/5/2007	Not Started	(2) Normal	
Acceptance Testing	Shaping Futurez Members	1/8/2007	1/10/2007	Not Started	(2) Normal	
Project Complete		1/11/2007	1/11/2007	Not Started	(2) Normal	

Figure 5: SharePoint task list

After evaluation of both JIRA and SharePoint task list and also completing our Interviews and Literature review the features in 5.1, were identified as a full set of requirement for our prototype.

5.1 Requirements List

(R1) Drag and Drop Tasks
Ability to move task notes between individual columns such as "Not Started", "In Progress", "Completed", etc.

(R2) Ability of users to add Comments
Users should be able to add comments to each others task.

(R3) Progress Bar for each tasks
The Note for each task should contain a progress bar displaying how much progress has been made on each task.

(R4) Automatic Assignment
The tasks in the "Not Started" column should be automatically assigned to the user dragging it to the "In Progress" column.

(R5) Details View
Users should be able to View the details of each task on a separate window which can be closed and opened if necessary.

(R6) Velocity calculation and end date projections
Calculate and Display the Current velocity and expected end dates of the project

(R7) Sprint filters
Implementing specific sprint filters, allowing users to choose specific sprints and displaying tasks belonging only to those specific sprints

(R8) Sprint planning
Allow users specifically project managers to plan each sprint by setting and modifying sprint information such as Start date, End date, Scrum master, etc.

(R9) Time reporting
Users should be able to report the amount of time spent on each individual tasks allowing project managers to calculate the total time spent on each project.

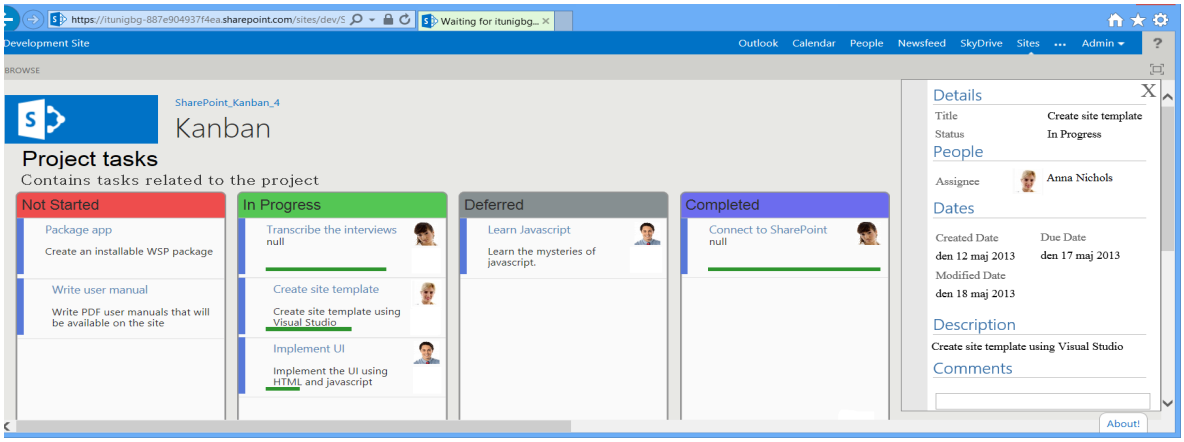


Figure 6: Kanban View App

(R10) Geographical project progress reports
Separate Project progress reports of different groups at individual geographic locations.

(R11) Chat feature
A chat feature allowing users to communicate between Project members.

(R12) Picture Sharing
Allow members of the each project to upload pictures or documents and share it between themselves.

(R13) Shared Drawing Board
An extension to the chat feature that would allow members to draw pictures to assist in explaining features/functionality in the project.

(R14) Automated task translation with manual approval workflow
A feature that automatically translates task titles and descriptions into a configurable number of languages, typically only to English. The translations are not visible until a user manually checks, optionally revises and finally approves the translated texts.

Once the requirement specification of our prototype was complete it was time to come up with the architecture of our prototype and decide on which framework to use in order to develop the tool. Accordingly we decided to use the SharePoint framework to develop our prototype as a SharePoint App, which would be an extension to the current SharePoint task list. This was firstly because it was a requirement from Mogul to use the SharePoint Framework in order to develop our prototype. Secondly, since the current SharePoint task list lacked most of the features we considered as a requirement, the tool would be quite beneficial to the users of SharePoint task lists.

We used an MVC design pattern in our architecture because we had a separate data and UI layer. The users interaction with the prototype would be managed by the controller, which would intern update the view depending on the users input. Most of the classes, their methods and attributes were identified and used in the architecture but some special features such as Chat were not included as there was not adequate time to implement those features.

6 Requirements Suggested by Practitioners

Using the right tools might mitigate some of these challenges. There are several tools in use at Mogul for the purpose of communication between members of distributed teams, such as email, Lync, Skype and videoconferences.

"You have Skype communication always and it's more difficult to feel responsible because they don't face the customer [...] even if you are in a video conference you feel a bit more responsible when you are showing the customer each use story. You feel more responsible and you do a better job because you test it, if you want to show it yourself", said one of the project mangers.

Using videoconferences improve communication compared to using text chat or writing email. Face-to-face interaction conveys extra information since the participants to read each other's body language and micro expressions.

A project manager said:

"I think it's very good if we use video conference or another tool. When we have daily meetings with the other part, we always try to use videoconference because you get a bit closer when you can see each other. You can understand if the

other person doesn't understand you at all […] It's more difficult with Skype chat or Skype"

6.1 Task lists

4 out of 6 interviewees have used task lists in SharePoint. They are not using them often anymore after JIRA has been implemented since they are limited and hard to interpret the progress of a project by looking at task lists.

"I think it's very basic. It's like one step up having all the tasks in an Excel file", said one of the developers.

Also, another developer stated:

"It is very limited so working with task we could open, read, modified and save"

One of the project managers pointed out:

"It is ok for simple tasks but also there is always way to improve it and make it better. But in general it is useful because you can easily track your tasks, you can enter how much you did so far, to enter status's of tasks, to re-assigning and so on. For the basic purpose it is ok but for any advanced things I would think it should have been broader. Something specific made for projects"

6.2 Virtual task boards

Customer collaboration is an important part in the Agile software development process. Hence, tools for task management will not only be used by the development team, but also by representatives from the customer. While this is possible with both task lists and virtual task boards, the lack of visual information presentation in task list systems make them far less intuitive.

Two developers said:

"They use JIRA. They have usually and we want them to have access to JIRA because they can see what's happening and they can go in and comment things, add information, and they can close tasks when they are done testing them"

"All customers are using JIRA during development to get in touch with the progress of the sprints and with use cases and the user stories. The customer can access it and see it and make comments and so on"

All developers and project managers at Mogul are using JIRA all the time. It seems delivering projects would be impossible without this tool.

"We use JIRA in all our projects, everything we do actually so nothing happen in Mogul without JIRA", said one of the project managers.

Also, one of the offsite team members said:

"We are using JIRA for everything. It is a great tool with great flexibility"

7 Prototype

To develop our SharePoint Kanban view App we used several different languages including HTML, CSS, and JavaScript. We also made use of the JQuery UI library to implement some of the features of our App. Visual studio 2012 was used for code editing and TFS for source code control. We also used office365 developer accounts in order to deploy and test our app.

The following is the GUI (see Figure 6) and explained below are how the different features that were implemented as they were thought to be the most important and relevant:

A. Draggable and Droppable Notes/Tasks

The feature was implemented by first reading the data from SharePoint task lists. Then task cards are displayed for each task in the task list. The JQuery UI library and its sortable list were used since the task cards needed to be draggable and droppable between columns.

The columns are individual divisions and each contained an unordered list and the task cards are list elements contained inside the unordered list. The task cards contain several individual divisions containing task title, description and user avatar. These unordered lists were connected as sortable lists in order to make it possible for the tasks to be dragged and dropped between the different columns.

B. Progress Bar

In order to implement the progress bar on each task card, JQuery UI library was used. The progress value for each task was acquired from the data retrieved and then used to create and display the progress bar.

C. Details View for each Notes

This feature was implemented by having a separate column that contains all the information details for any specific task card. The column was added every time a task card was clicked with all

the information related to the specific task including any comments.

D. Comments

The task comments were stored in a separate list located inside the app and it's not accessible by users except by using the Kanban tool.

8 Prototype Result

Even though Kanban style task management tools are very useful and powerful, not all companies have it. Some companies are still using list based task management.

"[We use task lists] a lot. […] We are working with team minutes in Lotus Notes and we are using it to create tasks, meetings and so on, so this could actually be quite an interesting way of also maybe use during meetings because then you have all your tasks in the meeting", a team leader said.

Having the task board interface make task lists more useful and more suitable for agile projects. But it is obviously not enough to use only task lists according to our interviews result.

It will however, improve collaboration and communication in agile processes compare to SharePoint task lists.

"It could [improve collaboration] […] we don't use task lists because we don't have any Kanban board"

"Yes [it could improve collaboration]. I think one reason is that you have built on a task in SharePoint, and if you have task list, I think you can connect it to outlook and you can view them in traditionally way, so definitely", an application manager said.

Out of the features present in the prototype, the most appreciated ones were *drag and drop* and *automatic assignment.*

The additional features that the most desired was *velocity calculation and end date projections* and *time reporting.*

8.1 Implemented Features

We decided focus on those features above (See Table 2) because those are the basic ones, which are more likely to give an impression to the people who try it.

The features of our prototype were based on the findings of our study. We tried to solve issues such as Task Management, Project Delays and Communication.

There were several different issues involved in task management such as physical distance making it increasingly difficult to communicate about task management or project issues (Herbsleb & Moitra, 2001). Inappropriate work distribution such as inexperienced developers being assigned more complex tasks while experienced developers working on less complex tasks (Hajratwala, 2012) is one such issue that would be solved by the use of several features of our prototype as shown in Table 2.

Project Delays were an important issue found during the literature review. Inadequate communication between different stakeholders or improper execution of synchronous agile planning meetings (Wang, et al., 2010) is one key issue causing Project Delays. But by the use of several features in our prototype including Progress Bar, Automatic Assignment, and Drag and Drop features (see Table 2) can help reduce the risk of these issues.

Communication is one of the biggest challenges in distributed ASD. Which is the reason why we specified the requirements for our prototype in such a manner so that it addresses the issues related to communication. Herbsleb & Moitra mentions Time-zone difference as one of the causes of miss-communication (2001), e.g. a company with distributed offices in United States and Thailand might face issues with the significant time difference. Decreased visibility of Project status or progress (Phalnikar, 2009) is another example of communication challenges, to deal with this issue we implemented a Progress bar for each specific tasks. In fact, most of the features implemented in our prototype were addressing the challenges of communication as shown in Table 2.

Some of the unimplemented features, such as *velocity calculation, time reporting, chat*, etc. would also solve several issues. The feature "automated task translation with manual approval workflow" was conceived specifically to address the issues related to language, combining the time saving of machine translation with a manual approval step to ensure that the translation is accurate. However the challenges with customer facing could not be solved by any of the mentioned features. This is due to the fact that it

	3.1 Task Management	3.2 Project Delays	3.3 Team Issues	3.4 Communication	4.1 Language	4.2 Customer-facing	4.3 Geographical distance barriers
Implemented Features							
Drag and Drop	X	X		X			
Comments				X			
Progress Bar	X	X		X			
Automatic Assignment	X	X					
Details View				X			
Future Features							
Velocity Calculation		X					X
Sprint Filter & Planning	X	X					
Time Reporting	X						
Chat Feature			X				X
Picture Sharing				X			
Shared Drawing Board				X			
Automated Task Translation with manual approval workflow				X	X		

Table 2: Challenges and Features to address them

must be addressed through frequent face-to-face meetings and discussion with the different stakeholders involved as mentioned by the interviewees.

In the sections below we explain in a detailed manner the implemented features of our prototype.

8.1.1 Drag & Drop Task Cards

The main goal of virtualizing the task board is to provide a computerized version of it. It is important that the features of physical task boards are not lost in the digital implementation. The two most important advantages of task boards are the overview that it provides and the ease with which statuses of tasks can be changed by simply moving the task card to a different column. In order to preserve this benefit in the digital Kanban tool, "drag & drop" was implemented so that users can simulate the moving of the task card by "dragging" the task card on the screen to a different column.

This feature addresses the "Task Management" problem in GSD projects by eliminating the need to manually replicate the changes between physical task boards in different locations.

The "always-updated" property of digital task boards can also contribute to reducing the risk of project delays caused by team members waiting for other members' tasks to complete before they can start working on the next task assigned to them since they can choose a different task instead and be certain that no one else has started working on it.

8.1.2 Automatic Assignment

Each team member can assign tasks to themselves by dragging and dropping task cards. When they move an unassigned task card from the first column to any other column, it will automatically be assigned to themselves. It also prevents the project delays since users don't need to wait for other members to complete required tasks since they can choose a different task by themselves in the mean time. This feature promotes the agile concept of self-organizing teams (Hoda, et al., 2013).

8.1.3 Progress Bar

Each task note has a progress bar that shows the progress of the task. Through this, it is very easy to see how many percentage of the task is completed. Also, users are able to quickly see how all the tasks are progressing in the overview, which will help the entire team to follow the overall progress of the project.

8.1.4 Details View

When a user selects a task card on the board, a detailed view of the task is displayed. Through this, users are able to see the title and the status, the name of assignee, created, due and modified date and the description of the task that was selected so that users can get all necessary information in one view of each task.

This will improve communication since users can get all information about the task without referring to the requirement specification or asking someone about it, which also mitigate the problem of time wastage, and ensures that the latest information about the task is always available in a single, shared location.

8.1.5 Comments

Comments are implemented in the details view so that users can easily communicate with the entire team about any task so that they can go back to the discussion history to check why the certain decision were made. This will improve communication since the discussion of each task is available to all users at any time.

9 Discussion

9.1 Perceived Benefits of the Prototype

Judging from the very positive reactions we were met with during the interviews after having demonstrated the prototype, it is clear that the task management implementation using lists leaves much to be desired. The ability to work visually with tasks by dragging and dropping them between columns was especially appreciated as it offers a way to manage and work with multiple tasks in a way that just isn't possible with task lists in their existing tool.

One of the interviewees stated that he would definitely like to implement it in his work processes *as is* even though the Kanban board app was only a prototype. This is a strong indication that the task board view addresses a very real and clear need.

Implementing the prototype as a layer on top of existing functionality gave a slightly unexpected advantage that was identified during the interviews: Since the task information is still stored in task lists in the existing system, the tools for analyzing task statistics already present in the platform can still be leveraged. As an example, the list of tasks can be exported to spreadsheet applications that in turn can be used to generate reports.

Further tie-ins into other collaboration and communication software were also mentioned as one of several desired features when the interview subjects started to realize the potential of the app. At the same time, it was pointed out to us that one of the attractive features of the prototype was that it was so simple to use, underlining the importance of keeping the user interface easy to understand and use.

All interviewees agreed that the Kanban view would make the underlying platform better suited for managing tasks in Agile development projects, however, the resistance against distributed development we were met with made it difficult to assess to what degree the tool would facilitate Agile processes in GSD projects. While there was no doubt that it *would* be a definite improvement, the sentiment was that the Task board view by itself is not sufficient to make global ASD project as efficient as performing development projects in collocated teams.

9.2 Strength and Limitations

The main strength of this paper is that it shows that adding a Kanban view to an existing system, leveraging technologies processes and infrastructure already in use, becomes a very useful tool for agile projects in distributed teams.

The result of our research is limited by a tight schedule that didn't allow for a greater number of interviews or a more fully-fledged Kanban application prototype.

A potential threat to the external validity of our findings is that only interview subjects from two companies were included in our interviews. An attempt to mitigate this risk was made by selecting interview participant from as many different roles as possible, and by keeping the focus of our research away from company specific issues to the greatest extent possible. One way we tried to address threats to internal validity in our research was by selecting interview candidates at random. A request for taking part as an interviewee in our research was sent out via email, to all the agile practitioners at Mogul and the people who were the first to respond to this request were interviewed to gather data for our study.

9.3 Future Research

After having discovered how GSD tools can be tailored to better support agile distributed processes, it would be interesting to study the use of the mentioned technologies in other industries not related to software development because the task management systems such as the one included in Microsoft SharePoint are not only used for software development projects. We recommend an in-depth study with empirical data on the difference in project success ratios depending on whether a virtual task board was used or not, to further confirm our results. Lastly, further research about other ways that agile processes can be adapted to a distributed environment could be also important in order to better reconcile these two trends in software development.

10 Conclusion

Based on the responses we received during the interviews with people experienced in ASD and list based task management after having demonstrated our task board prototype, we can conclude that implementing virtual task boards in GSD tools improves communication between geographically distributed team members and reduces the risk of project delays in distributed ASD projects.

Moreover, additional advantages can be achieved by implementing the virtual task board on top of existing collaboration and document management platforms.

In answering our research question stated in the introduction we have identified the following features required for a successful implementation of a virtual task board:

- Drag&Drop of task cards
- Automatic task assignment
- Progress indicators
- Detailed information view
- Comments

11 Acknowledgements

Carrying out this Bachelor Thesis was indeed an interesting and rewarding experience. We would like to thank Mogul Gothenburg especially our supervisor, David Litmark, for his patience and guidance, Robert Fridén for his expert advice and Philip Nestenborg for giving us the opportunity to carry out our thesis at Mogul. We are also thankful to IT University of Gothenburg and its professors, especially Lars Pareto for all the ideas and support that makes this research possible.

References

Aitken, A. & Ilango, V., 2013. A Comparative AnalysTraditional Software Engineering and Agile Software Development. 2013 46th Hawaii International Conference on System Sciences, pp.4751–4760. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6480417 [Accessed April 24, 2013].

Battin, R.D. et al., 2001. Leveraging resources in global software development. Software, IEEE, 18(2), pp.70–77. Available at: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=914750&queryText%3DLeveraging+Resources+in+Global+Software+Development [Accessed April 19, 2013].

Booth, W.C., Colomb, G.G. & Williams, J.M., 2009. The Craft of Research, Third Edition, University of Chicago Press. Available at: http://books.google.se/books?id=Y31pUtkwb2oC.

Dingsøyr, T. et al., 2012. A decade of agile methodologies: Towards explaining agile software development. *Journal of Systems and Software*, 85(6), pp.1213–1221. Available at: http://linkinghub.elsevier.com/retrieve/pii/S0164121212000532 [Accessed March 1, 2013].

Guang-yong, H., 2011. Study and practice of import Scrum agile software development. In: Communication Software and Networks (ICCSN), 2011 IEEE 3rd International Conference on. pp.217–220. Available at: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=6013698&queryText%3DScrum+in+software+development[Accessed May 25, 2013].

Hajratwala, N., 2012. Task Board Evolution. Agile Conference AGILE 2012, pp.111–116. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6298100 [Accessed April 19, 2013].

Herbsleb, J.D. & Moitra, D., 2001. Global software development. Software, IEEE, 18(2), pp.16–20. Available at: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=914732&queryText%3DHerbsleb+and+Moitra [Accessed May 25, 2013].

Hevner, A.R. et al., 2004. Design science in information systems research. MIS Q., 28(1), pp.75–105. Available at: http://dl.acm.org/citation.cfm?id=2017212.2017217 [Accessed April 19, 2013].

Hoda, R., Noble, J. & Marshall, S., 2013. Self-Organizing Roles on Agile Software Development Teams. *IEEE Transactions on Software Engineering*, 39(3), pp.422–444. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6197202 [Accessed March 1, 2013].

Kajko-Mattsson, M. et al., 2009. Long-Term Perspective of Agile Methods. 2009 Fourth International Conference on Software Engineering Advances, pp.1–2. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5298422 [Accessed May 25, 2013].

Kamaruddin, N.K., Arshad, N.H. and Mohamed, A., 2012. Chaos issues on communication in Agile Global Software Development. In: Business Engineering and Industrial Applications Colloquium (BEIAC), 2012 IEEE. pp.394–398 Available at: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=6226091&queryText%3Dcommunication+issues+in+agile+software+development [Accessed May 25, 2013].

Liskin, O. & Schneider, K., 2012. Improving Project Communication with Virtual Team Boards. 2012 IEEE Seventh International Conference on Global Software Engineering Workshops, pp.35–36. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6337316 [Accessed April 19, 2013].

McFeeley, R., 1996. *IDEAL: A Users Guide for Software Process Improvement.* [online] Available at:<http://www.sei.cmu.edu/library/abstracts/reports/96hb001.cfm> [Accessed April 19, 2013].

Mohagheghi, P., 2004. Global software development: Issues, solutions, and challenges, Retrieved April 07, 2009. Available at: http://www.idi.ntnu.no/grupper/su/publ/parastoo/gsd-presentation-slides.pdf [Accessed April 19, 2013].

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. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4777400 [Accessed March 24, 2013].

Sahay, S., Nicholson, B. and Krishna, S., 2003. Global IT Outsourcing: Software Development Across Borders. [online] Cambridge University Press. Available at: <http://books.google.se/books?id=G2X42-YOiY8C>[Accessed April 24, 2013].

Vaishnavi, V. & Kuechler, W. , 2004. Design Science Research in Information Systems. [online] Available at: <http://www.desrist.org/design-research-in-information-systems/> [Accessed April 19, 2013].

Wang, X. et al., 2010. Tools for Supporting Distributed Agile Project Planning. In D. Šmite, N. B. Moe, & P. J. Ågerfalk, eds. Agility Across Time and Space SE - 13. Springer Berlin Heidelberg, pp. 183–199. Available at: http://dx.doi.org/10.1007/978-3-642-12442-6_13[Accessed April 19, 2013].

Appendix A (Interview Questions)

1. Can you tell us about your current responsibilities in the company?

2. Could you us give a picture of how much do you practice Agile methodologies in the company or in your current project (Which Agile methods are you using e.g. Scrum, Kanban, Crystal Clear, etc)?

3. Can you tell us how widely are the resources distributed geographically in your current project (Development teams, managers, departments, offices etc)?

4. Do you use SharePoint?
   - Have you used task lists in SharePoint? If so, what do you think about it? (follow-up)

5. Do you think Agile processes have any benefits by using task lists in SharePoint?
   - If not, why? If yes, how could it be improved? (follow-up)

6. Could you briefly describe the situation relating to the use of Global Software Development (GSD) tools in the project you are involved in?

7. What are the main challenges in working Agile in a distributed environment?
   - From your perspective, which of these challenges have the greatest impact and how?
   - How do you think the use of GSD tools can help in overcoming those challenges? (follow-up)

8. Do you feel a part of the team even while working distributed? If not how do you think the situation can be improved?

9. In your opinion, what are the shortcomings in the current GSD tools you are using?

10. What are the necessary/important improvements in current GSD tools you expect in order to improve Agile processes?

11. What could you advice to any other company using GSD tools in order to improve their Agile processes?

12. Do you have any suggestions on how JIRA can be improved?

13. Is there anything more you would like to add?

7. Do you see any other potential uses for the Kanban app besides agile software development projects?

8. How would you rate the usefulness of the following features in the Kanban view (from 1 to 5)?
    1. Comments
    2. Progress bar
    3. Drag and drop
    4. Automatic assignment

9. How would you rate these potential features? (from 1 to 5
    1. Velocity calculation and end date projections
    2. Sprints filter
    3. Sprint planning
    4. Time reporting
    5. Graphical project progress reports
    6. Chat feature

Appendix B (Interview Questions about our prototype)

1. Could you tell a little bit about your current responsibilities?

2. In your work, do you ever use task lists in SharePoint?

3. In your opinion does the Kanban app make task lists more useful? In what way?

4. In your opinion, what would you suggest as features which could be added in future to this Kanban app in order to make it more useful?

5. Do you use agile in your current projects? Would you say that the Kanban board makes task lists more suitable for agile projects?

6. Do you think the use of this app can improve collaboration and communication in the agile processes compare to SharePoint task lists?