UNIVERSITY OF CALGARY

Distributed AgilePlanner: A Card-Based Planning Environment for Agile Teams

by

ROBERT ENRICO MORGAN

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE

DEGREE OF MASTER OF SCIENCE

DEPARTMENT OF COMPUTER SCIENCE

CALGARY, ALBERTA

JANUARY 2008

UNIVERSITY OF CALGARY

FACULTY OF GRADUATE STUDIES

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies for acceptance, a thesis entitled " Distributed AgilePlanner: A Card-Based Planning Environment for Agile Teams" submitted by ROBERT ENRICO MORGAN in partial fulfilment of the requirements of the degree of Master of Science.

Supervisor, Dr. Frank Oliver Maurer, Department of Computer Science

---

Ron Murch, Haskayne School of Business

Thomas Zimmermann, Department of Computer Science

Date

Abstract

Supporting distributed agile teams as they plan a project is challenging. When collocated, teams use index cards for the creation and organization of stories during a planning meeting. In a distributed setting using these index card isn't convenient as some of the team members do not have access to the physical cards. This thesis presents Distributed AgilePlanner, a card-based distributed planning tool for agile teams. Distributed AgilePlanner aims to combine the benefits of physical card-based planning with those of existing online tools. The tool supports interacting with story cards and is comparable with interacting with physical cards. To evaluate Distributed AgilePlanner a qualitative evaluation was conducted with two software development teams based in academia and three teams simulating the software development process. In addition, an external informal evaluation was conducted. Feedback was positive with participants excited about using the tool and its potential.

Acknowledgements

Throughout my journey working on this research many have helped and encouraged me along the way. I would like to take this opportunity to thank everyone that helped and supported me.

To Dr. Maurer, Thanks for all your help, guidance and support over the past few years. I don't think I could have made it without your continued support and encouragement.

To the internship students who helped in the development of DAP. Thanks for all your help and for continuing to work on DAP following this research.

To my fellow researchers, Carmen, Xueling, David, Patrick, and Chengyao thanks for being so analytical and for letting me interrupt you as often as I did to bounce ideas off you.

To my family, thanks for helping me and supporting me throughout my entire education.

Dedication

To Stephanie, who supported me no matter what.

Publications From This Thesis

Materials and ideas from this thesis may have previously appeared in the following peer-reviewed publications.

Robert Morgan, Jagoda Walny, Henning Kolenda, Estaban Ginez and Frank Maurer: Using Horizontal Displays for Distributed & Collocated Agile Planning, Proceedings of the 8th International Conference on Agile Processes in Software Engineering and eXtreme Programming (XP 2007), Como, Italy 2007 (Springer).

Robert Morgan, Frank Maurer: MasePlanner: A Card-Based Distributed Planning Tool for Agile Teams, Proceedings, International Conference on Global Software Engineering (ICGSE 2006), IEEE Computer, 2006.

Table of Contents

List of Tables

List of Figures and Illustrations

Chapter One: Introduction

Software development is highly dependent on effective communication. The success of many software projects is affected by the teams' ability to effectively communicate ideas and requirements among the customers, their representatives and the developers. In teams that follow agile practices, meetings are scheduled where all parties involved can come together and plan the next stage of the project. Bringing the entire team together every few weeks is not always possible when the team is distributed around the country or around the world. Verbal communication plays an important role in this planning. In agile teams requirements are typically represented by information written on paper index cards. Sharing these paper index cards with distributed members during the meeting is next to impossible. Providing support for sharing the content of these index cards during the planning meetings is what motivates this research.

1.1 Agile Project Planning

Agile methods are a group of methodologies that have similar beliefs to developing software. The main principle of agile methods is to "… satisfy the customer through early and continuous delivery of valuable software" [AgileManifesto 2001]. This means that agile teams do their best to communicate effectively with each other while trying to deliver working software as frequently as they can [AgileManifesto 2001].

Teams following the agile principles work closely with their customers, often communicating face-to-face. Distributed project planning presents challenges to the agile teams in that fact-to-face communication is no longer possible. Teams using agile

methods in a distributed setting need to adjust how they communicate in order to continue to satisfy their customers.

Planning in an agile project involves developers, project managers and customers to gather together and plan out the direction of the project. Planning in an agile project is iterative with meetings typically scheduled every 2-4 weeks [Beck 2000]. In these planning meetings, customers are responsible for creating stories. Stories are requirements that bring business value to the customer; they typically do not include technical details. In collocated environments, these stories are written on paper index cards (3.5 x 5.5 inches) and placed on a table or a storyboard for everyone to see and interact with.

Once the stories are created, developers are responsible for estimating the amount of effort needed to complete each of the cards. Often, developers will break a story card into smaller cards in order to better understand or describe the requirements [Beck 2000] before estimating them. The way in which estimations are collected varies from team to team with some teams using hours and others using some other means of estimation, sometimes referred to as story points [Cohn 2006]. The estimation can also vary with some teams providing one estimate value; while others provide a best case, worst case and most likely estimate. These values are used to help the team both schedule the cards and prioritize them. Actual effort for the stories is also recorded following the completion of a given story (usually prior to the planning meeting) and can be used for a number of different purposes: burn down charts, velocity, or simply to help developers better estimate future stories [Cohn 2006].

Scheduling of story cards is done via iterations. A main component of agile project planning is the use of an iterative development cycle. Iterations are fixed time boxes whereby the scope (i.e. the number of stories) is changing [Beck 2000]. Customers prioritize stories to assist in planning the iteration. Ideally, customers will rank the stories that have the most value to them first with subsequent stories having less value. This process involves negotiating with the developers as some stories, might represent broken functionality that needs to be resolved before work can continue or in the case where some stories are pre-requisites to others.

To assist with planning, teams determine the available effort to help guide how many story cards can be included in the iteration. Available effort, like estimates, depends on how effort is recorded. If teams are using hours, then available effort could be the number of developers multiplied by the number of hours that each developer is working on that project in a week [Cohn 2006]. Available effort can assist when teams are negotiating the scope of the iteration as it provides another indicator to determine whether if the stories can be completed in the time allotted.

Stories that are not scheduled in the current iteration can tentatively be scheduled in a future iteration, to be discussed at the next planning meeting, or can remain unscheduled. Unscheduled stories are said to be in the product backlog, or just backlog, and are available to be scheduled at a later date. The same planning process is repeated for every planning meeting.

1.2 Motivation

The motivation behind my research involves distributed agile teams as they conduct project planning. Information pertaining to planning data is important for understanding the direction of the project. The information written on the index cards need to be shared with all members of the team in order for everyone to be able to fully contribute to the project planning. In addition to this written information, the implied information that can come from a story's proximity to the other cards and the way in which they are organized on the table/board is important. The problem arises when some team members do not have access to this planning information during the planning meetings. Without access to the index cards or planning surface, distributed team members become dependent on others for their understanding of the project plan. This can lead to misunderstandings or miscommunications between team members as information from the story cards, for example, is summarized or changed as one team member reads or explains the story to the other individual.

Existing tools attempt to support sharing the information contained within these story cards. However, these tools are primarily webpage based and follow the approach of the user/web browser pulling updates of the page from the server when they're ready (or automatically at regular intervals). The second issue with existing systems lies within the way information is presented to users. Most systems use a tabular presentation approach. This does not really provide a way for teams to group related stories together, thus losing the information that comes with a given card's proximity to other story cards. The third issue with existing tools is the way team members interact with the planning artefacts. Interacting with paper index cards is quick and intuitive; the same cannot be

said of many existing online planning tools. Online planning tools tend to require users to follow a specific flow to create or edit artefacts. This can be as simple as clicking or tabbing through multiple fields in a table to being as complex as navigating away from the webpage to enter information and then returning to the previous page. Other tools require users to click on a submit button before the changes are saved and able to be shared with others.

In trying to better understand the planning issues plaguing a distributed team, I observed a group of developers conducting planning meetings where the customer was located in another country. The planning occurred with the development team creating story cards and organizing them on a table and making use of a speaker phone for verbal communication with the customer. Following the planning meeting, the development team would spend anywhere between five and fifteen minutes entering the stories into an online system in order to make the information available to the client.

This approach always placed the customer at a disadvantage in terms of access to information. Customers would have to wait until after the meeting to check that the stories they had explained and were expecting were in fact the stories that were understood and posted by the rest of the team. During my observations the teams discussions were very thorough and misunderstandings regarding stories rarely occurred. The delayed sharing of story information runs the risk of a having one team expect one thing and the other expecting something else. This type of misunderstanding occurred once during my observations where the developer's understanding of the story was different from the customer's. The misunderstanding was minor and easily resolved following a discussion at the following planning meeting. Miscommunications need to be

avoided as the distributed nature of the team seems to cause less communication with the customers following the planning meeting. This was seen during my observations where almost no communication occurred between the developers and the customer during the iteration and in the rare cases where communication did occur it was done via email and resulted in time delays as responses were sent between customer and developer.

1.3 Research Problem

The main problem for this research is to determine how to better support card based planning for distributed agile teams and to determine if tools that support card based planning are effective.

1.4 Goals

In order to solve the research problem a card based planning tool that supports distributed agile teams needed to be developed. The goals of the tool and this research are as follows:

- Support creating and organizing planning information,
- Support real-time updating of planning information,
- Provide an intuitive planning environment,
- Provide a means for nonverbal communication,
- Support interactions that are similar to interacting with physical index cards (natural interaction),
- Evaluate the tool to determine if it better supports distributed teams.

1.5 Thesis structure

The remainder of this work is presented as follows: Chapter Two provides background information on existing research in the area of agile planning tools and in the area of groupware. Chapter Three presents a list of requirements that are necessary for distributed agile planning tools. Chapter Four looks at the design and implementation of the proof of concept tool, Distributed AgilePlanner. Chapter Five compares the various tools, including Distributed AgilePlanner, in their ability to satisfy the requirements presented in Chapter Three. Chapter Six provides an initial qualitative evaluation of Distributed AgilePlanner. Chapter Seven concludes this work.

Chapter Two: Related Works

Software systems to support agile project planning (and general collaboration) in a collocated and distributed environment have been available for some time. The options available for distributed collaboration vary from sharing an Excel [Microsoft 2007:Excel] spreadsheet (via email or remote desktop), to using wiki pages [Wiki.org 2002] all the way to custom agile planning tools such as: [Rally 2007, Danube 2007, VersionOne 2007]. The existing agile planning tools all provide team members with basic functionality to create, edit and discard stories. In addition, they also provide some means to organize the stories into iterations. The limitation that many of these tools exhibit is their inability to truly support interacting with the stories in a way that presents the stories as cards (card-based planning) and allows interactions that are similar to physical interactions with index cards (natural interaction). The majority of existing tools also have failed to consider recommendations from the human computer interaction (HCI) community. This chapter will present a summary of existing agile planning tools (focusing on custom tools) followed by a summary of the research from the HCI's groupware community.

2.1 Collaboration Tools

Collaborative tools like Microsoft NetMeeting [Microsoft 2007: NetMeeting, Wikipedia 2007:NetMeeting] have been used for sharing desktops and traditional single user applications like Excel [Microsoft 2007:Excel]. Sharing one's desktop environment ensures that all participants are able to see the same information and that there is no duplication of information. Using spreadsheets for story planning makes it easy to rank

stories by dragging the row higher or lower in the sheet. The limitations to this means of interacting are that individuals who do not have control need to ask for it (different from face-to-face planning) and once the planning meeting is over the shared sheet needs to be distributed to all the participants. This sharing of the spreadsheet now runs the risk that the information could be modified by someone and not re-shared with the others thus you could have inconsistencies [Rees 2002].

Another popular choice for group collaboration is a wiki system [Wiki.org 2002] A popular example of this type of collaboration is Wikipedia [Wikipedia 2007:Wikipedia]. These web-based systems provide individuals with the ability to edit the content of the pages with little effort and share them with anyone. This approach is hindered due to the inherent nature of web pages in general; the page must be refreshed for the new information to be displayed.

Both wiki and spreadsheet based planning can be effective however they are not specific to nor do they have workflows that are explicitly designed for agile planning.

2.2 Agile Planning Tools

Specific tools for agile planning look strictly at supporting agile teams and have adjusted their workflow as such. The focus of the tools varies from one to the other; however, the bulk of the tool focus is more on story based planning where the priority is to present the story data in any format and not on card based planning where the priority is to present story data on an index card that can be easily moved, altered, stacked, etc. The existing tools can be categorized into three groups: Form-Based, Hybrid Form and Wiki Based, and Board-Based planning systems [Liu 2005].

2.2.1 Form Based Planning Systems

Form based planning systems provide the bulk of existing planning tools and are typically web based systems that use web forms to enter and represent planning information. Existing form based planning systems include commercial products like Rally [Rally 2007], VersionOne [VersionOne 2007] and ScrumWorks [Danube 2007] and open source products like XPlanner [XPlanner 2007] and Moomba [Reeses et.al. 2004]. These systems use forms to present the planning information like iterations and stories (Figure 2.1). In addition, they make it easy to derive supplementary information such as total estimates for a given iteration or the ranking of a story. Form based systems provide all the basic functionality that is necessary for planning a project (create, edit, delete, prioritize) in addition to searching for content and sorting information based on a given field [Liu 2005]. Their greatest limitation is that they are restrictive in how individuals and teams are able to interact with them and in the way that the user interface is updated.

2.2.2 Combined Wiki and Form Based Planning Systems

As we have seen, wiki's are popular ways of sharing information with others and they provide flexibility with respect to the content. Combining this with the benefits of a Form-Based planning system provides agile teams with a much more flexible tool for planning. Tools like MASE [Maurer 2002, EBE 2007] are examples of a combined planning tool that lets teams create stories and organize them like Form-Based planning systems and also lets teams annotate the story's wiki page with information that does not

fit in the form (Figure 2.2). Once again, this approach has limitations. Wiki systems are still dependent on the user to initiate an update, or have their browser auto update to receive the latest information. They do not allow for the project data to determine when an update is necessary.

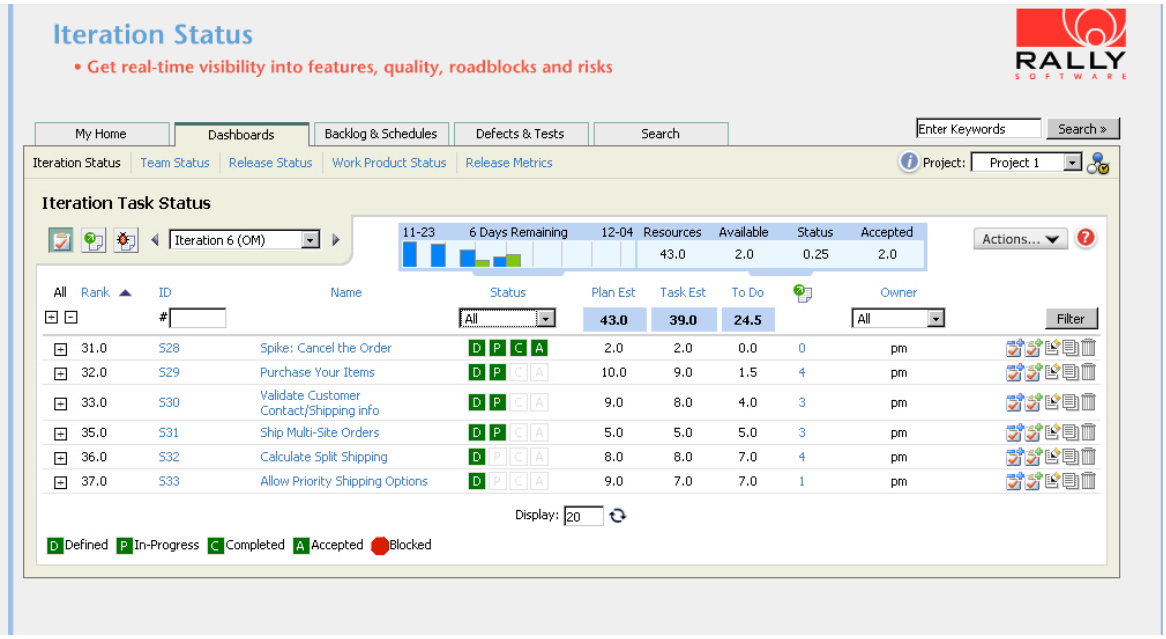


Figure 2.1: Rally Planning Interface [Rally 2007]

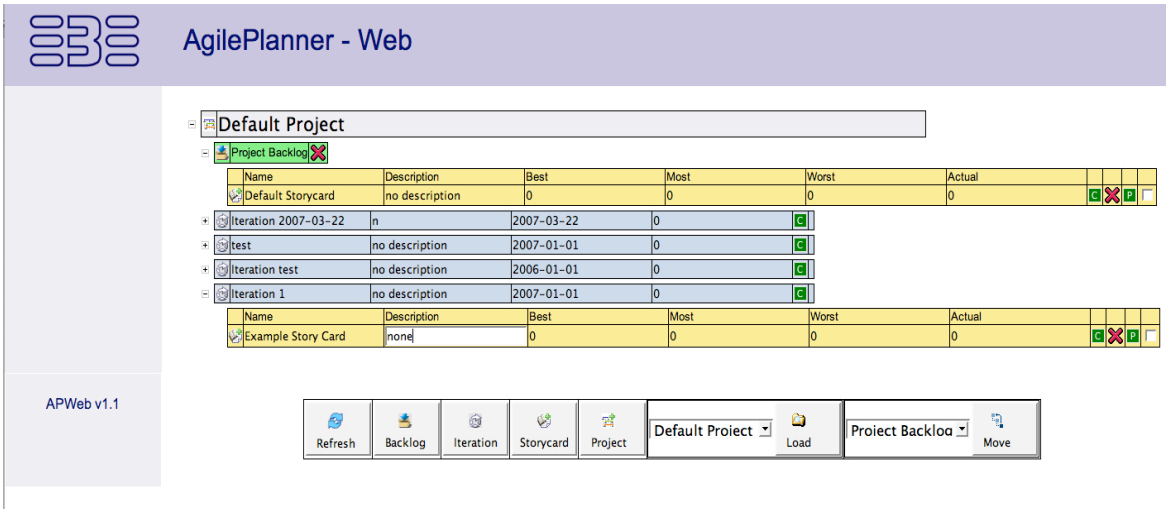


Figure 2.2: Mase Planning Interface [EBE2007]

2.2.3 Board-Based Planning Systems

Board-Based planning systems are systems that use visual representations for planning artefacts [Liu 2005]. These types of systems move towards mimicking physical card based planning. There exists a few systems that fall into this category: Glue Wiki [Floranta 2007], Mingle [Thoughtworks 2007], CardMeeting [CardMeeting 2007] and AgilePlanner [Liu 2005]. The benefit that Board-Based systems bring to planning is that they allow teams to interact with the cards.

CardMeeting (Figure 2.3) uses two-dimensional representations of index cards that teams can enter text upon and organize like paper index cards. It uses a browser to allow team members from any location to interact with the cards. Individuals are able to create cards, organize them like they would paper index cards and edit their text. In addition, CardMeeting is updating the planning environment in near real-time to allow for better synchronization between distributed team members. However, it doesn't

provide any of the advantages that form based planning tools provide, such as summarizing estimates and ranking.

2.2.4 Limitations of Existing Planning Systems

Both Form-Based and Form and Wiki-Based systems suffer from a number of limitations compared to face-to-face planning meetings. The first limitation is that they do not provide any information regarding the relationships among cards. In physical planning meetings, cards that are related can be grouped together to imply a relationship. This spatial awareness [Gutwin et.al. 1996] can contain important information about the project. The second limitation comes from the lack of knowledge of who is interacting with the planning artefacts and participating in the planning meeting.

Board-Based planning systems rectify the spatial awareness limitation. However, existing tools [CardMeeting 2007, EBE 2007, Floranta 2007, Rally 2007, Danube 2007, XPlanner 2007, Thoughtworks 2007] do not provide any concessions to determine who is participating in the planning and to a greater extent who is interacting with the planning artefacts. The knowledge regarding who is participating in the planning and interacting with a planning artefact is important as it encourages communication and therefore collaboration [Gutwin et.al. 1998].

Figure 2.3: CardMeeting [CardMeeting 20007]

2.2.5 Agile Planning Tool Support and Requirements.

Research into distributed agile methods provides some insight into the requirements for tool support. The basic features necessary to conduct planning are: creating stories and iterations, editing the content of these artefacts, organizing the stories and ranking them. Previous examinations into planning tools for agile teams have identified some required features to truly support distributed agile planning.

Developments by Schummer et.al. [Schummer et.al. 2001] in TUKAN highlighted some key points when developing collaborative software for agile teams. Schummer's [Schummer et.al. 2001] subsequent examination of the XP planning game found that the team members' interactions with planning artefacts were an important

source of information for the others present. They also concluded that cards (and planning artefacts) need to be remotely accessible to everyone, and that there needs to be a way for team members to communicate with each other. The issue of awareness among team members was also brought up on more than one occasion.

The Moomba tool [Reeses at.al 2004], and subsequent research, took many of the recommendations that were made by Schummer et.al. [Schummer et.al. 2001] and extended it to provide more collaboration aspects such as providing users with a relaxed WYSIWIS (What You See Is What I See) interface. This approach, from groupware research [Stefik et.al. 1987, Greenberg 1990], provides users with the opportunity to collaborate within the system more effectively. WYSIWIS is looked at more in depth in Section 2.3.1.

Providing support for communication, coordination and cooperation in a distributed XP team is challenging. In Ketler et.al's [Kelter et.al 2003] examination of how tools support agile teams, they found that tools supporting communication, coordination and cooperation tend to: support synchronous and asynchronous communication; provide shared access to documents and allow for joint editing of said documents; allow for planning and managing schedules; and finally, provide electronic representations of physical objects [Schummer et.al. 2001].

A summary of requirements necessary for the development of a digital planning environment is presented in Liu [Liu 2005]. In this work, he identifies seventeen functional and interaction related requirements for building planning tools (See Table 2.1). The specified interaction requirements, focused predominantly on supporting agile planning in a face-to-face environment using digital tabletops and omitted some

requirements for building tools to support distributed group interactions. The following section (Section 2.3) focuses on requirements for building distributed group application, called “groupware”.

Criteria	Criteria Sub Categories	Existent Agile Planning Support						
		Physical Planning Media	XP Planner	VersionOne	RallySoft	Wiki	MASE	GlueWiki
Agile planning objects creation and editing	Functional Requirements	√	√	√	√	√	√	√
Visual characteristics for different types of story		√	√ / X	√ / X	√ / X	√ / X	√ / X	X
Agile planning metrics management		X	√	√	√	√	√	√
Planning for multiple iterations		√	√	√	√	√	√	X
Systematic organization of planning objects		X	√	√	√	√	√	X
Fluid transition between plan changes and the consequent results		√	√ / X	√ / X	√ / X	√ / X	√ / X	√ / X
Concept of team and identity authentication		√	√	√	√	√	√	√
Real-time exposure of the plan via the Internet		X	X	X	X	X	X	√
Simultaneous planning information organization	Interaction-related Requirements	√	X	X	X	X	X	X
Simultaneous planning information editing		√	X	X	X	X	X	X
Story editing with handwriting inputs		√	X	X	X	X	X	X
Handwriting recognition capability		X	X	X	X	X	X	X
Fluid transition between individual and collaborative work in agile planning		√	X	X	X	X	X	X
Fluid transition between agile tabletop collaboration and external work		√	X	X	X	X	X	X
Flexible user arrangement		√	X	X	X	X	X	X
Shared access to digital objects in agile planning		√	X	X	X	X	X	X
Use of physical planning objects		√	X	X	X	X	X	X

Table 2.1: Assessment for existing tool support for agile planning [Liu 2005]

2.3 Groupware

Research involving collaboration on a computer has been around for quite some time. A large body of research in the Human Computer Interaction (HCI) community has made significant strides in understanding how to better support collaboration using computers. One area of research in HCI, called Groupware, involves improving understanding of how a software application can improve "…support for groups of individuals working together towards a common goal" [Grudin 1994]. Much of the research in the Groupware or collaboration aware community [Ellis et.al.1991, Tang 1991, Ishii et.al. 1993] has focused on generalized group tasks and the literature tends to focus on more generic tasks, such as drawing or editing text [Ellis et.al 1991, Tang 1991]. This has allowed in depth examinations of factors influencing group collaboration and the compilation of a large set of recommendations for groupware systems. The recommendations are typically generalized to groupware applications and the relevancy to planning varies. However, the body of research is important to consider and a summary relevant to planning is presented here.

Early observation done by Tang [Tang 1991] produced a significant number of recommendations pertaining to the design of groupware applications. He concluded that the most important things to consider when building groupware are: people use hand gestures when communicating, the process of creating artefacts is as important as the artefact, the workspace is an important tool to mediate collaboration, and spatial orientation plays a role in structuring the collaborative activity [Tang 1991].

Reinhard et.al. [Reinhard et.al. 1994] looked at developing a taxonomy to evaluate groupware tools. The taxonomy is broken down into three main components: Application criteria and requirements, Functional criteria and requirements, and Technical criteria and requirements. All of these components are evaluated based on their flexibility in terms of the user's ability to dynamically switch between the states of the application [Reinhard et.al. 1994]. Of particular interest are the functional criteria and requirements as these tend to look at issues specific to collaboration-aware application and distributed groupware applications. Specifically, visualization of data was highlighted as an important point. The What You See Is What I See (WYSIWIS) paradigm [Stefik et.al 1987, Greenberg 1990] for how shared data is presented to all parties is of particular interest. Reinhard et.al [Reinhard et.al. 1994] briefly looked at the pure WYSIWIS vs. the relaxed variants and looked at their flexibility. They concluded that a more relaxed WYSIWIS approach is more flexible because it allows users to create customized views for sharing and presenting data [Reinhard et.al. 1994]. (see section 2.3.1 below for more on WYSIWIS).

However, given the recommendations from Tang and the taxonomy for evaluating tools, Groupware applications are still not regarded as having a good track record. Grudin [Grudin 1994] looked at why groupware applications tend to fail and concluded that their failure can be narrowed down to three problems in their design and evaluation. Grudin states that the disparity between those who perform the majority of the interactions with the application and those that benefit from it tends to be too great, and that there needs to be more balance among both the workload involved and the benefits. The example that he provides for this is an online calendar system for managers and workers (who don't

use computers on a daily basis). The bulk of the work is for the workers as they have to keep their calendar up to date but this is more of a benefit for the management staff so that they can schedule meetings [Grudin 1994]. Second, there is a breakdown of intuitive decision-making due, in part, to the lack of understanding of those driving the development of a groupware application and the intended user population. This involved software project managers driving the development of a tool for a field for which they have no in-depth knowledge [Grudin 1994]. This results in parts of the application being neglected or features, valuable to other user categories, being deemed unimportant. Lastly, he points out that the difficulties in evaluating groupware tools are due to the difficulties in creating "…a group in the lab that will reflect the social, motivational, economic and political factors that are central to group performance" [Malone et.al. 1992] and the fact that evaluation of groups requires more people using the tools and longer observation periods. In all three cases, Grudin [Grudin 1994] identifies that a smaller and more homogeneous target population can simplify the development and can add to the success of the tool.

Mandviwalla et.at [Mandviwalla et.al. 1994] conducted an interdisciplinary literature review to develop a set of generic requirements for developing groupware applications. The result of their review produced a set of requirements (Table 2.2) that they suggest be considered within the context of the project when developing groupware applications. They also found that even when the context of the intended tool is well defined, tool flexibility is important for its adoption. Flexibility in supporting different ways that groups can interact with a tool is important due to the impact that group

dynamics and composition will have on the individual experiences with the tool [Mandviwalla et.al. 1994].

Generic Groupware Design Requirements	
GR1	Support multiple group tasks
GR2	Support multiple work methods
GR3	Support the development of the group
GR4	Provide interchangeable interaction methods
GR5	Sustain multiple behavioural Characteristics
GR6	Accommodate permeable group boundaries
GR7	Adjustable to the group's context

Table 2.2 Generic Groupware Design Requirements [Mandviwalla et.al.1994]

Groupware applications can generally be categorized into two different categories: shared windowing systems, (input/output is shared amongst multiple clients/users without the application being aware or changed to handle multiple clients/users) and collaboration aware application (applications built specifically for group usage) [Lauwers et.al. 1990]. In Lauwers et.al. the examination of shared windowing systems highlighted a number of limitations focusing around input, output and start-up consistency with most of the limitations being resolved by creating a custom collaboration-aware application. As many of the limitations of shared-windowing systems become resolved with collaboration aware applications, the remainder of this section will focus on this research area.

2.3.1 Workspace Awareness

An important part of groupware applications is workspace awareness. Gutwin [Gutwin et.al. 1998] looks at workspace awareness in the context of real-time groupware applications and explains it as: "… one of the keys that allows people to interact in and through the shared workspace." [Gutwin et.al. 1998]. In essence, this allows others to be able to see where an individual is working and what they are doing. There are three parts to workspace awareness: knowing what the others are able to see (their viewport), knowing where their mouse pointer is located and when it is moving, and finally seeing the movements of artefacts in the workspace [Gutwin et.al. 1998].

The three components of workspace awareness were incorporated by Gutwin et.al. [Gutwin et.al. 1998] to provide a radar view for a real-time groupware application. They compared the radar to a simplified overview of the application in which the remote user interactions were not displayed. They observed individuals performing three different tasks: a follow task, a direct task and a copy task. Their results found improvements in the performance for the direct and follow tasks but did not see any change in the copy tasks. This led them to derive three lessons for groupware application development. Firstly, radars are good for providing information regarding: orienting one's self in the application, navigating through the application, understanding the global state, and to carry out individual tasks. Secondly, providing the workspace awareness information in the radar can improve speed, efficiency and satisfaction. However, their third lesson learned is that providing a radar view does not automatically improve performance of the given tasks and the ability for a group to work together will have a direct impact on the group's performance. The observations made are important to group

collaboration, however, tasks relating to artefact organization and creation, central to agile planning, were missing.

A reoccurring requirement that comes from both groupware and agile literature revolves around the WYSIWIS paradigm. WYSIWIS is an effective way of representing and visualizing shared data. The way WYSIWIS is implemented can vary from strict, where the visuals are identical, to varying degrees of relaxed, where the individuals have more flexibility. Pure WYSIWIS can limit the amount of independent work that individuals are able to carry out and forces everyone's focus onto the same shared information. In its relaxed form, individual users are able to customize their view of the shared information so that individual work can be accomplished [Stefik et.al. 1987, Greenberg 1990, Reinhard et.al. 1994]. However, the amount of flexibility will be related to how relaxed the WYSIWIS is implemented. When tasks require users be able to see what others are doing but not necessarily at the time they are working on the data, Reinhard [Reinhard et.al. 1994] recommends a relaxed approach.

In face-to-face situations, workspace awareness is augmented by our ability to see others point and gesture with their hands. In a digital environment it is the mouse pointer and our ability to see the mouse pointer's location and movements that provides this spatial awareness. In order to share this information in real-time, groupware systems remote mouse pointers, or telepointers as they are often referred to, are displayed in the shared workspace. Telepointers have been the focus of some research in the groupware community and specifically their use has been examined for their workspace awareness benefits and gesture support [Greenberg 1990, Gutwin et.al. 1996, Gutwin et.al. 1998].

Examinations by Dyck et.al [Dyck et.al. 2004, Gutwin et.al. 1998] at developing high performance telepointers is motivated by the lack of telepointers in existing commercial groupware applications and their poor performance in research application. Dyke cites that poor treatment of telepointers leads to their poor performance and that they must be treated as streaming media in order to be successful [Dyck et.al. 2004, Gutwin et.al. 1998]. The research looked into developing high performance telepointers by producing a framework for developers in addition to some conclusions on telepointer implementation. The strategies examined found that supporting low bandwidth needs and minimizing data loss could not be accomplished by a given strategy and that telepointers implemented using the TCP network protocol are typically unsuitable for real-time implementations due to the high overhead preventing data loss.

The second benefit that telepointers provide is the ability to support gestures in a collaborative environment. Gesturing can be important in providing context to, and augment, the conversation being had. Gutwin et.al. [Gutwin et.al. 1998] found that by providing telepointers as part of the shared workspace awareness mechanism, that questions regarding the collaboration environment, such as: who is working in the environment and where they are working, can easily be answered. (See Table 2.3 for a full list).

Element	Relevant Questions
Presence	Who is participating in the activity?
Location	Where are they working?
Activity Level	How active are they in the workspace?
Actions	What are they doing? What are their current activities and tasks?
Intentions	What will they do next? Where will they be?
Changes	What changes are they making, and where?
Objects	What objects are they using?
Extents	What can they see? How far can they reach?
Abilities	What can they do?
Sphere of Influence	Where can they make changes?
Expectations	What do they need me to do next?

Table 2.3: Elements of workspace awareness [Gutwin et.al.1996]

2.3.2 Verbal Communication

A very important aspect to any groupware application is the support for verbal communication. Existing studies [Greenberg 1990, Ellis et.al 1991, Maddviwalla et.al. 1994] indicate that, without a clear voice communication channel, the supporting features, like gestures and other non verbal communication mechanisms, are not as effective. The importance of including a clear voice channel as part of the groupware application is not stressed in the literature. The ability to use existing telecommunication systems, such as traditional telephone systems or Voice-over IP systems like Skype [Skype 2007], as opposed to incorporating this into the groupware systems, appears to be adequate.

2.4 Summary

The existing body of work for supporting agile planning in a digital environment and building real-time groupware applications is substantial. This chapter presented an overview of existing research and tools developed for collaborative work and agile planning. We can see that existing planning tools provide many of the basic functionalities necessary for creating, organizing, and sharing project plans. The bulk of the tools tend to fall short when it comes to ease of interaction that exists with physical card-based planning. This chapter's overview of groupware research highlights the relevant fraction of the knowledge gained by observing and building groupware applications. Specifically, we saw that workspace awareness is important to consider when building groupware applications and that a relaxed WYSIWIS approach combined with telepointers can go a long way in improving interaction.

Chapter Three: Distributed Planning Tool Requirements

Requirements for creating a distributed agile planning application come from the agile and groupware bodies of knowledge as well as my observations and experience with distributed agile teams. Tool requirements from the existing research, software tools and observations, can be broken down into those specific to agile planning and those specific to real-time groupware application development.

3.1 Agile Planning requirements

The agile planning requirements stem from the need to create, organize and share planning information. Existing planning tools, such as those presented above, all exhibit the fundamental set of required features for a planning tool. Based on the tools presented above and based on paper based planning, the following agile planning requirements can be derived:

1. **Creating, editing and deleting planning objects.** The fundamental requirement of any agile planning tool is its ability to support the creation, modification, and deletion of planning objects. Without this requirement, a tool would not be able to support agile planning. [Liu 2005, EBE 2007, Floranta 2007, Rally 2007, Danube 2007, VersionOne 2007, XPlanner 2007].
2. **Agile planning metrics management.** Experience working on a project can be an important piece of information when trying to plan for the future. Keeping track of knowledge from previous iterations and story cards, such as estimates, priority, and actual effort, can be useful when trying to determine

team performance, and can be of use when trying to estimate newer tasks (yesterday's weather [Beck 2000]). Managing this information can also be of use when negotiating the scope of current iterations. [Liu 2005].

3. **Planning multiple iterations.** Supporting multiple iterations when planning allows teams not only to plan at the iteration level but also to conduct long term release planning. It allows for improved story coordination between planning sessions. [Liu 2005, EBE 2007, Floranta 2007, Rally 2007, Danube 2007, VersionOne 2007, XPlanner 2007].
4. **Moving stories from one iteration to another.** Without supporting the ability to change what iteration a story card is part of, you are not able to plan effectively. [Liu 2005, EBE 2007, Floranta 2007, Rally 2007, Danube 2007, VersionOne 2007, XPlanner 2007].
5. **Team and identity authentication.** Security is important to prevent unauthorized access and modification to the information contained in the project plan. [Liu 2005, EBE 2007, Floranta 2007, Rally 2007, Danube 2007, VersionOne 2007, XPlanner 2007].
6. **Real-time exposure of the plan via the Internet.** This requirement provides remote access to the project data such that, as changes occur, updated information is available for access via a web interfaces or other means. [Liu 2005].
7. **Visual characteristics for different types of stories.** Stories can often be broken down into distinguishable types like bug fixes, new features, changes to existing functionality or enhancements to name a few. Supporting a visual

distinction between these different types of story cards can be beneficial to teams. [Liu 2005]

8. **Integration with the development environment.** Planning tools are used by both the business side and the technical development side of the team. Supporting integration with the development environment increases access to the planning information for developers. [Ketler et.al. 2003]

3.2 Groupware Tool Recommendations

Requirements and recommendations from the groupware community tend to focus on generic group tasks as a whole. It is important to note that much of the literature, in making groupware recommendations, indicates that some of the requirements will not apply to specific tasks and are presented as requirements only to force designers to consider their need when developing groupware systems [Mandviwalla et.al. 2004]. Based on the groupware literature presented, the following requirements can be derived:

1. **Fluid transition between individual and collaborative work.** Systems need to support some way of distinguishing private data from public data. In essence groupware tools need to allow a means of hiding data from others. [Mandviwalla et.al. 1994, Reinhard et.al. 1994, Liu 2005].
2. **Telepointers for pointing and gesturing.** Allow for remote mouse pointers to be displayed to show who is accessing what data and to allow for gesturing in the workspace. [Gutwin et.al. 1998, Dyck et.al. 2004].

3. **Flexible information sharing.** Through relaxed WYSIWIS, sharing information requires that changes to one workspace need to be updated in another's workspace. [Stefik et.al. 1987].
4. **Change notification.** When changes to the workspace are made those changes need to be shared with the other team members regardless if they are connected to the plan or not. [Stefik et.al. 1987, Kelter et.al. 2003].
5. **Radars for awareness information.** Knowledge of where others are working in the environment is important. An overview of the workspace that doesn't include awareness information is not sufficient. Radars share the awareness information in addition to the overview of the environment. [Gutwin et.al. 1998].
6. **Flexible participant management.** Team members should be able to connect and disconnect with ease and not affect others connected to the system. Support for adding new group members, or creating a new group are also necessary. [Reinhard et.al. 1994].
7. **Fluid subgroup formation and dissolution.** When supporting subgroup creation, the potential for isolation needs to be minimized in order for the subgroup to be informed of the other participants. [Stefik et.al. 1987].
8. **Simultaneous interaction.** Supporting team members interacting simultaneously is important as it better follows how teams interact in a collocated environment. Forcing teams to take turns would result in more overhead for the team and the planning. [Mandviwalla et.al. 1994, Liu 2005].

3.3 Summary

In combining my observations and experiences with distributed agile teams and the lessons learned, recommendations and requirements from existing agile planning tools, and groupware systems, presented in Chapter Two, a list of requirements for developing a real-time agile planning application was derived.

Chapter Four: Distributed AgilePlanner

Distributed AgilePlanner (DAP) is a real-time groupware application that provides agile teams with a digital environment for card-based planning. DAP is an attempt to fulfill as many of the requirements presented in the previous section as possible in order to build the best possible distributed agile planning tool that follows the card-based planning metaphor. The focus of this chapter is to present an in-depth review of the features supported in DAP, in addition to some implementation details.

4.1 Tool Overview

DAP provides a shared workspace where teams can create and interact with digital planning artefacts simultaneously and in near real-time (Figure 4.1). The artefacts are presented to the team in the form of two-dimensional objects on the display. The 2D representation provides team members with a means to organize and plan in a way that is similar to paper-based planning. The shared workspace allows distributed team members to interact with the planning artefacts without having to be in the same room.

Virtual artefacts provide an additional benefit in that the state of a project's plan can be saved and reconstructed at a later time. This allows teams to plan an iteration and, following the completion of the iteration, the team can then go back and review the contents of the previous iteration quickly and with little effort.

DAP supports three different types of artefacts within a project in a hierarchical structure (Figure 4.2): iteration, backlog and story card. The backlog and iteration objects are similar objects in that they act as containers for the story cards. The most notable

difference is that the iteration objects are used for scheduling story cards whereas the backlog is used to contain unscheduled story cards.

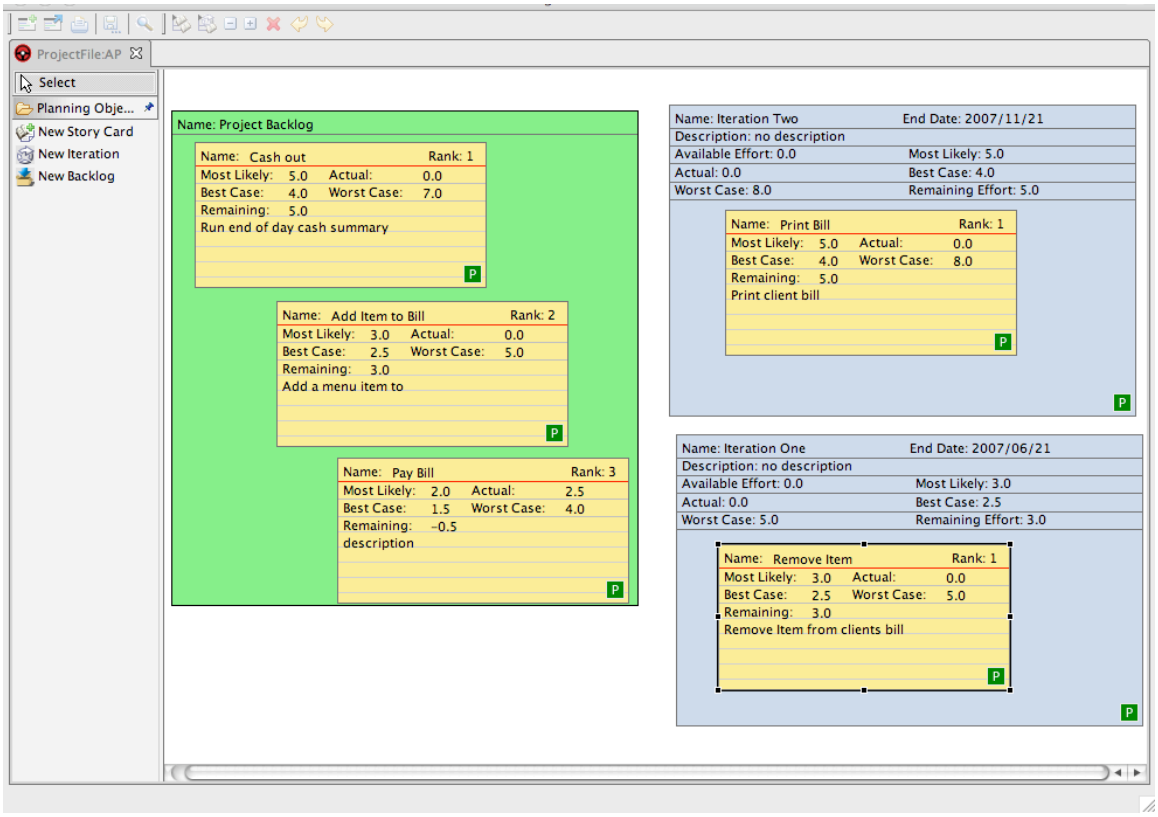


Figure 4.1: DAP Environment

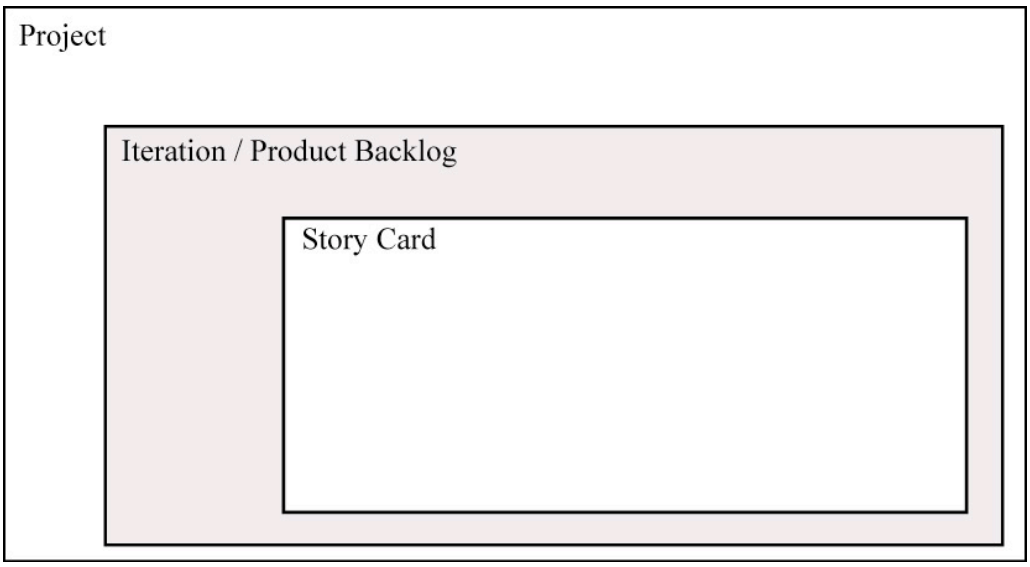


Figure 4.2: DAP Supported hierarchy

4.1.1 Creating Planning Objects

Creating planning objects is straightforward and attempts to limit the amount of learning required of an individual. The process of creating any of the three planning objects is exactly the same. First, the user selects the type of object that they wish to create and, using the mouse, clicks at the desired location to create the object (Figure 4.3). This approach to creating planning objects was chosen as it closely follows the idea of picking up a paper index card from a stack of blank cards and placing it on a table.

Iteration and Backlog objects differ from the story card objects when it comes to the location where they can be placed on the planning surface. The Iteration and Backlog objects are free to exist anywhere in the planning workspace. The story cards are limited to exist only inside the Iterations and Backlog objects. This restriction is enforced by the system to ensure that the hierarchical relationship between the planning objects is kept and well defined. The system handles this by changing the mouse icon when the create

story card is selected and cursor location is not overtop an iteration or backlog artefact (Figure 4.4).

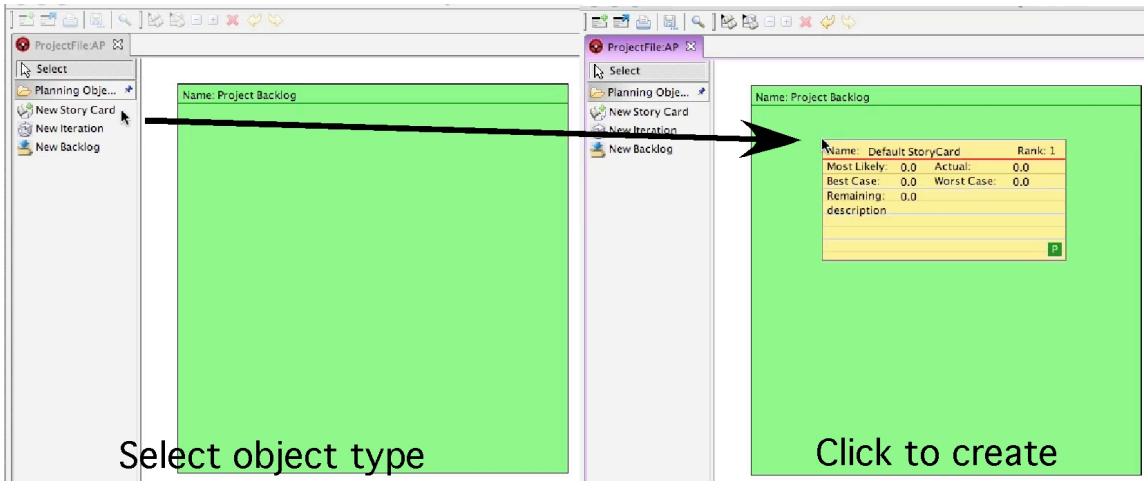


Figure 4.3: Creating a planning object

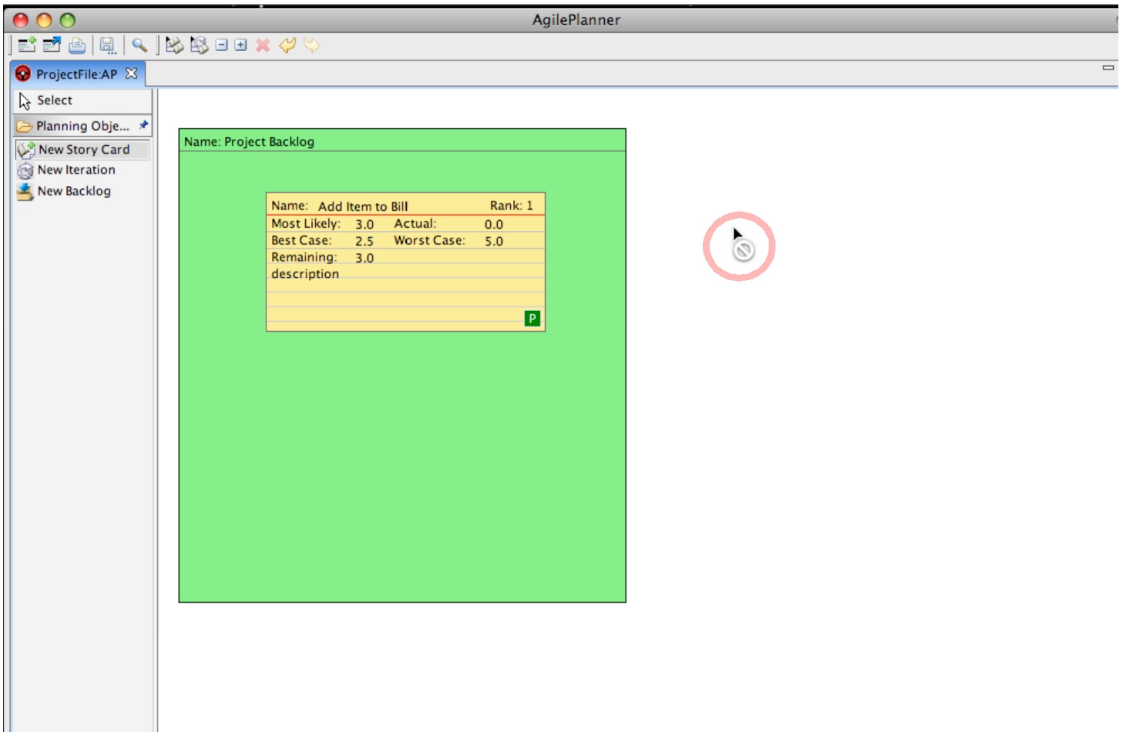


Figure 4.4: Mouse icon when new object is not possible

4.1.2 Editing Planning Objects

Editing of planning objects is handled by means of directly editing the text on-screen. The user selects the artefact to edit and clicks on the text they wish to change. This approach is inspired by the way that individuals would edit the contents of a paper index card. During my initial observations, the team members observed would grab a card from the planning surface, either cross-out or erase the contents that needed to be changed and add the new text. The closest digital equivalent, that supported a keyboard and mouse, was to select the card, then select the text to modify and change the text (Figure 4.5). The amount of information that is modifiable on a given planning object is dependent on the type of object (see Table 4.1 for a complete list of editable values).

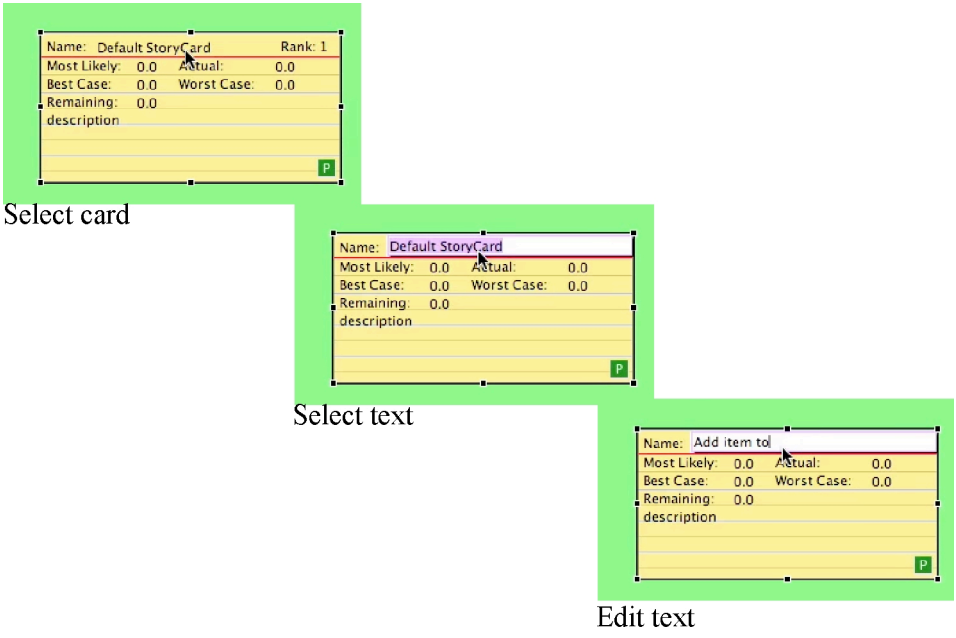


Figure 4.5: Selecting and editing text

Field Name	Story Card	Iteration	Backlog
Name	Y	Y	N
Description	Y	Y	N/A
End Date	N/A	Y	N/A
Best Case Estimate	Y	N	N/A
Worst Case Estimate	Y	N	N/A
Most Likely Estimate	Y	N	N/A
Actual Effort	Y	N	N/A
Remaining Effort	N	N	N/A
Available Effort	N/A	Y	N/A

Table 4.1: Editable Artefact Fields

4.1.3 Moving and Organizing

Moving and organizing cards are the functions in which DAP moves away from more traditional planning tools and closer to card-based planning. To make planning as easy and intuitive as possible, DAP uses drag-and-drop to interact with planning artefacts (Figure 4.6)

Moving story cards in DAP tries to follow the idea of picking up a physical card. In DAP the user clicks on the card, drags it to the new location and lets go of the mouse button to drop the card. One improvement that DAP provides over paper-based planning comes from moving Iterations or the Backlog that contains a given number of story cards. Iterations and the Backlog act as containers for story cards. This allows for easy movement of the entire Iteration/Backlog without the need to recreate the story card arrangement within.

In an environment using paper index cards, where cards are spread out on a table and grouped into Iterations and Backlog, moving an entire Iteration or Backlog often

involves moving a handful of cards at a time. DAP's use of Iteration and backlog objects alleviates this by having these objects act as containers for the contained story cards. Moving an Iteration containing story cards, for example, is as simple as moving the Iteration object (Figure 4.7). This allows teams to organize story cards in a way that best fits their needs and the location of the story cards in the iteration is kept when the Iteration is moved. Moving story cards within an Iteration/Backlog is no different than moving a story card between an Iteration/Backlog (Figure 4.8).

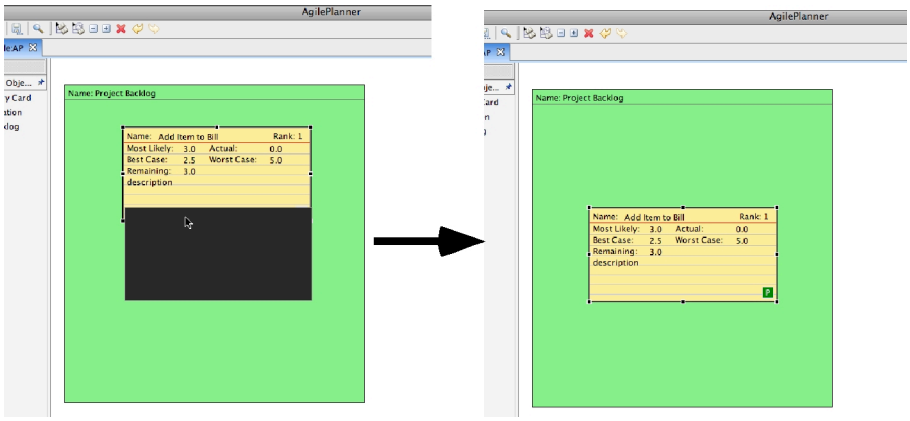


Figure 4.6: Drag-and-drop to move a story card

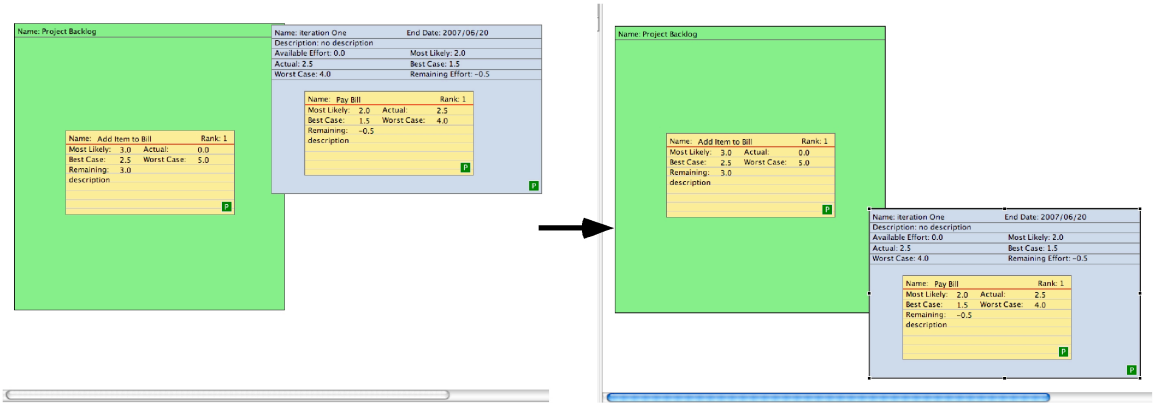


Figure 4.7: Moving an Iteration containing story cards

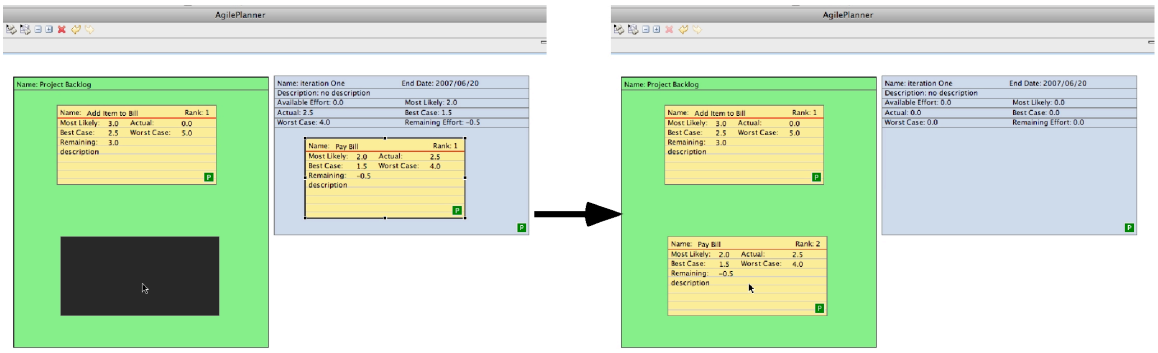


Figure 4.8: Moving cards between Iterations and the backlog

The way in which planning artefacts are organized in the Planning workspace can provide important contextual information for teams. Organizing the contents of the Planning workspace can be important to: denote ranking or priority for the story cards, indicate a dependency or relationship between two story cards, or can be simply used to utilize the available space. DAP attempts to accommodate users in how they wish to organize their workspace by being as flexible as possible.

Story dependencies, inter-story relationships and priority often use location or proximity as a key indicator. The proximity of a given card to other cards can be used for a variety of things such as: a cards' relationship to another, its order in terms of what card need to be completed before another or even the level of importance within a sub-group of cards. DAP ensures that organizing of cards in this way is possible by letting users have near total control in the organization of cards. Cards are free to be organized in any way that the user sees fit. The only limitation to this freedom is that story cards need to exist inside an Iteration or Backlog (Figure 4.9).

Prioritization of story cards is an important portion of many planning meetings and the location of cards is often used to imply the ranking rather than an explicit indicator. Story cards with the highest ranking are often placed at the top of a story board or at the top end of the table from where the customer is sitting. Currently DAP automatically ranks story cards in the workspace. The ranking system uses the location of the card with respect to the top of the Iteration/Backlog object to assign a rank (Figure 4.10). Changing a particular story card's rank is as easy as dragging it up or down in the Iteration/Backlog (Figure 4.11).

Screen real-estate can be a serious concern with a planning tool. In a physical environment planning space is less of a premium. Finding a room with a large table or board to use during the planning meeting is often not an issue. This is not the case with digital environments. The amount of visible space is limited by the resolution of the display being used. Therefore, a means of optimizing the use of the digital planning environment is necessary. DAP's approach to addressing this issue is through a mechanism that collapses story cards to show only the story name (Figure 4.12). This can save a considerable amount of space while keeping the most important information from the story cards visible. Double clicking on a given card will collapse that card, expanding the card again also uses the double click action. "Collapse All" and "Expand All" buttons are provided as a convenience.

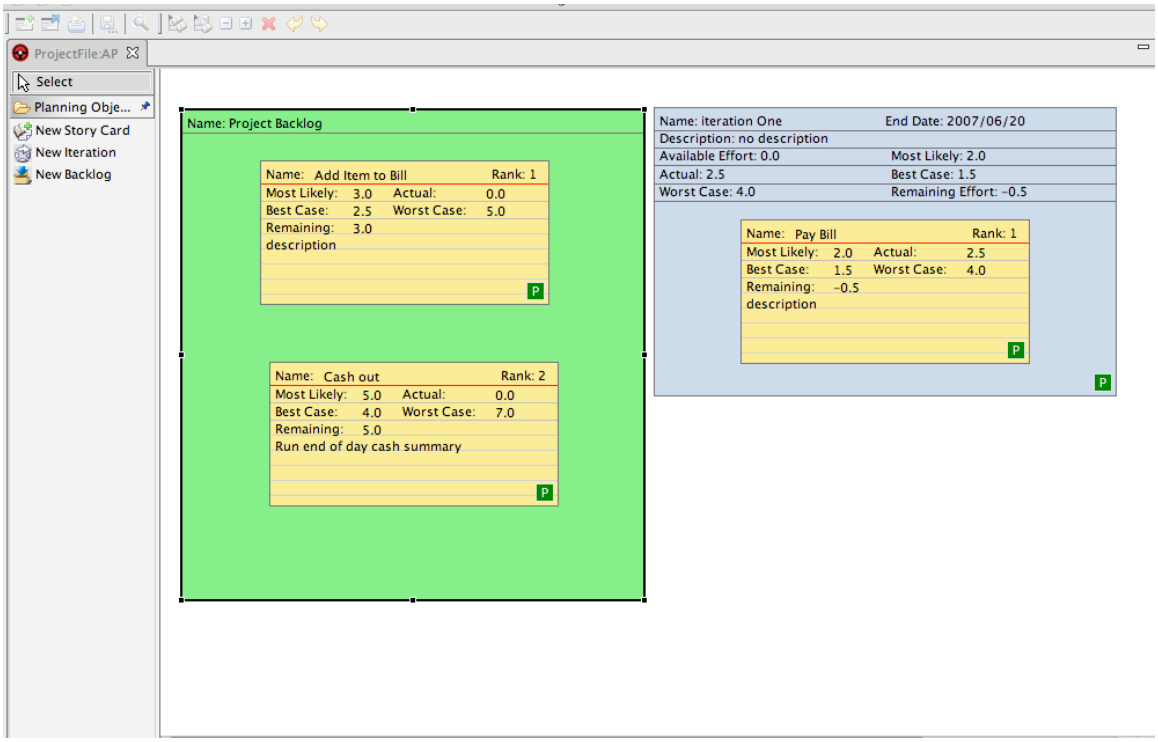


Figure 4.9: Organized story cards

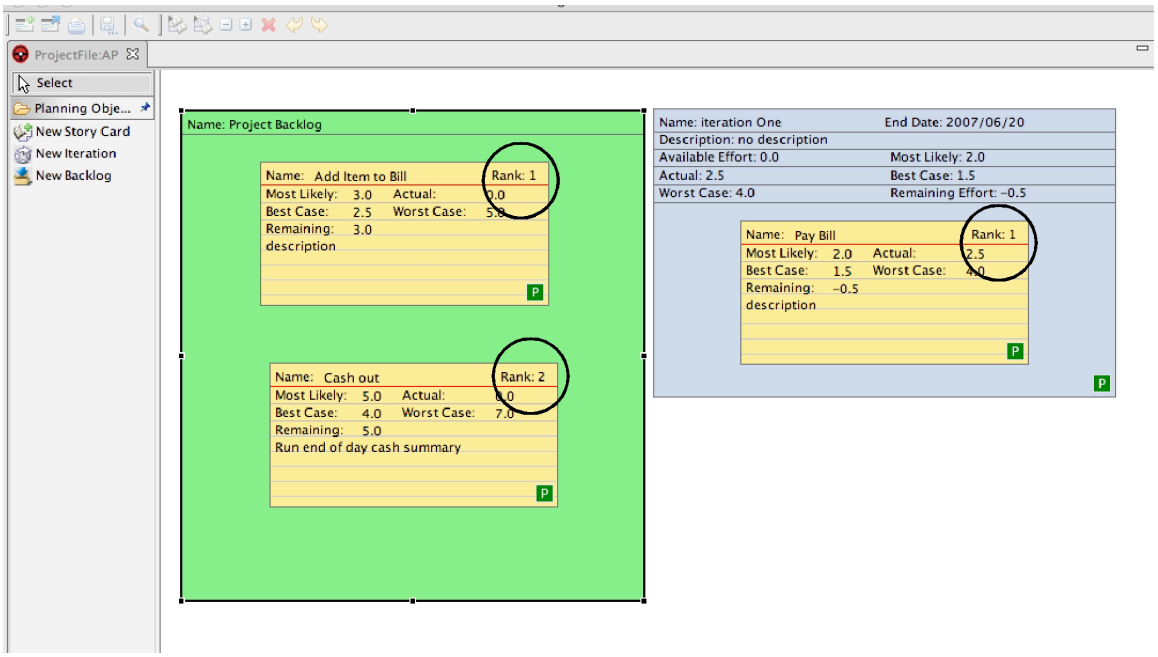


Figure 4.10: Ranking of story cards

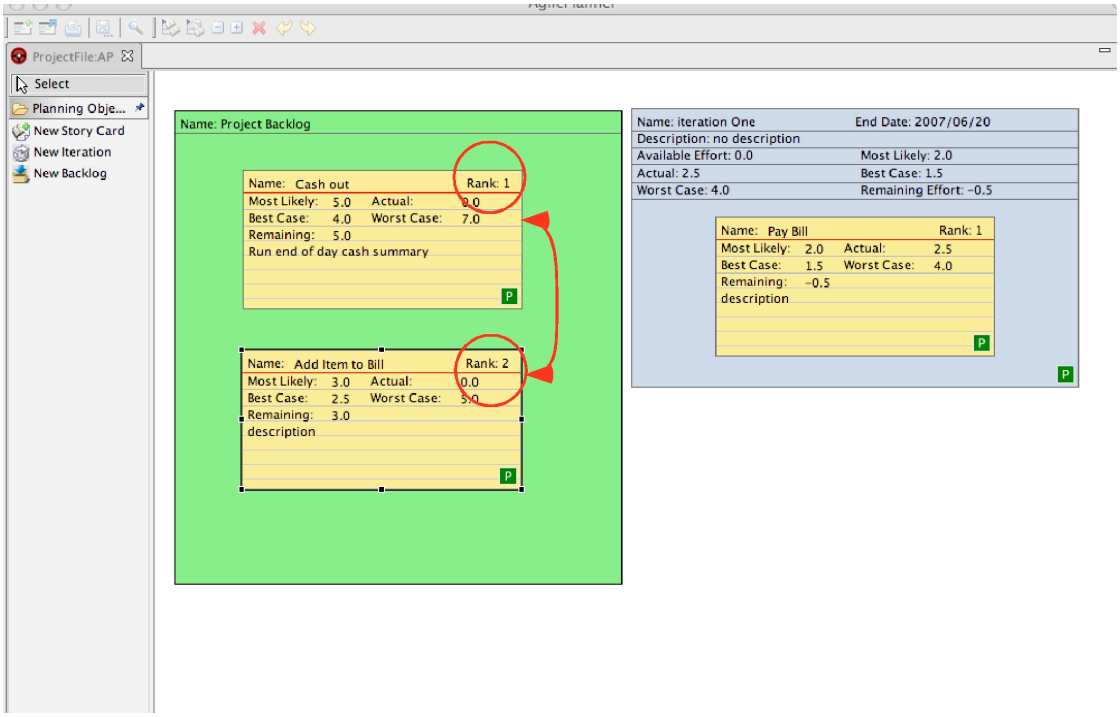


Figure 4.11: Changing a story cards rank

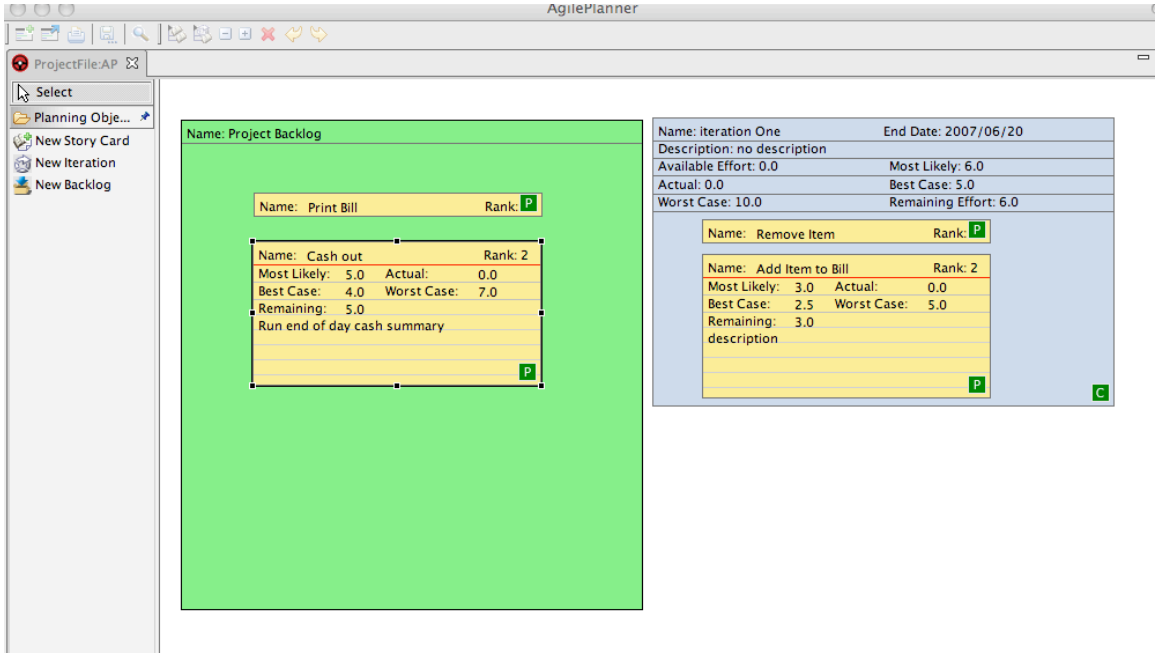


Figure 4.12: Collapsed/Expanded story card

4.1.4 Project Accounting

Accounting information like story points or hours can be an important aspect to effectively planning an iteration. This is a common feature in many of the existing agile planning tools; however, organizations differ in what information they collect. Some organizations only keep track of one estimate value whereas others keep track of a best case, worst case and most likely estimate values. DAP provides flexibility when it comes to keeping track of these estimate values.

DAP's approach to project accounting is to let the team decide what type of story accounting they desire. Teams can have a single estimate value for the stories or any combination of estimate values (Figure 4.13). In addition to the estimates, a remaining effort option is also available to provide teams at a glance how much of the estimated time is available for that story card. This information can then be used to help teams better estimate effort in the future.

Similarly to existing systems, DAP provides summary information at the iteration level to allow team members to get an idea of how much is estimated for the given iterations (Figure 4.14). This summary information is tied to the types of estimate information being displayed at the story level and is updated automatically when a story is added or removed or a contained story is changed. Iterations, like story cards, keep track of remaining effort. The remaining effort for the iteration is calculated based on the values from its contained story. This, in conjunction with the option to enter the available effort regardless of using story points or available time, can give teams an instant

indicator of how much room is available for additional story cards or if the team is on track to completing stories within the budgeted time.

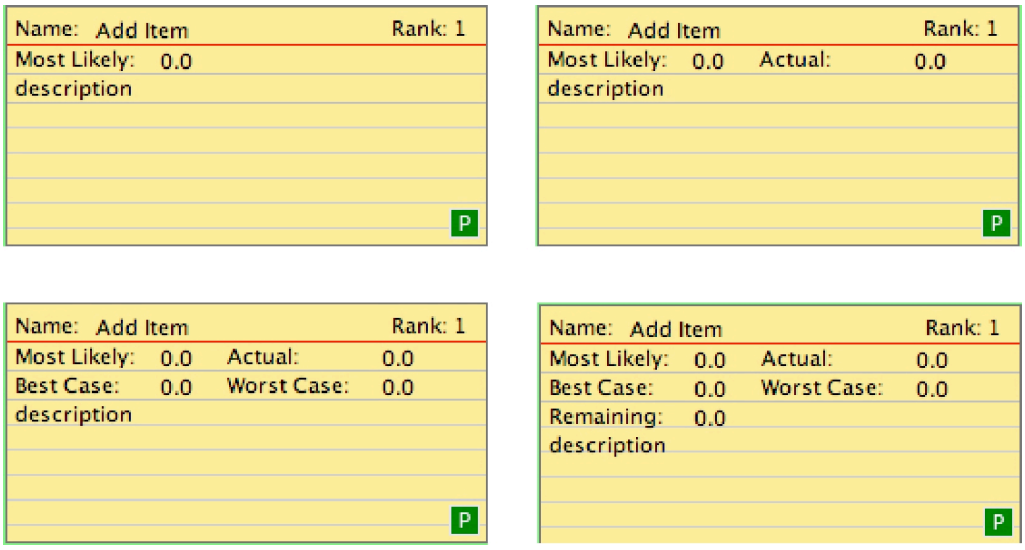


Figure 4.13: Estimate combinations

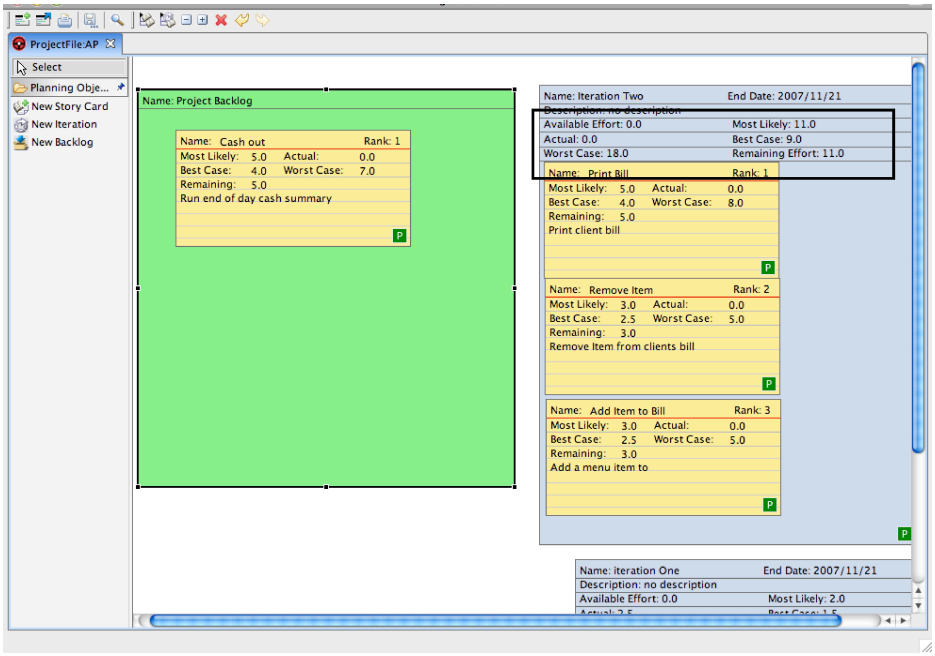


Figure 4.14: Iteration summary

4.1.5 Changing Status

Knowledge of whether a story is completed or not (in progress) is important to plan effectively in a project. Usually a story is not always completed at the end of an iteration and we cannot assume that once the iteration is over, all the contained stories are automatically complete. During the planning meetings using paper index cards, an incomplete card is easily moved to a new iteration or to the backlog depending on the discussion with the team. DAP provides this same flexibility, stories that are incomplete at the end of the iteration can be moved to a new iteration or backlog.

Marking stories as complete is another important factor to marking the progress of the project. This information can be important to various parties involved in the planning process (eg: project manager, customers). DAP uses a small indicator at the bottom of the story card and iteration to indicate the state of the story and iteration (Figure 4.15). Stories and iterations are the only objects that have a state that can be changed, as the backlog is always ongoing. Changing a story/iteration from in-progress to complete is accomplished by clicking on the indicator. In the case of the story cards, clicking on the indicator changes the card from in-progress to complete. In the case of the iterations, however, changing the status of the iteration will subsequently change the status of all contained story cards.

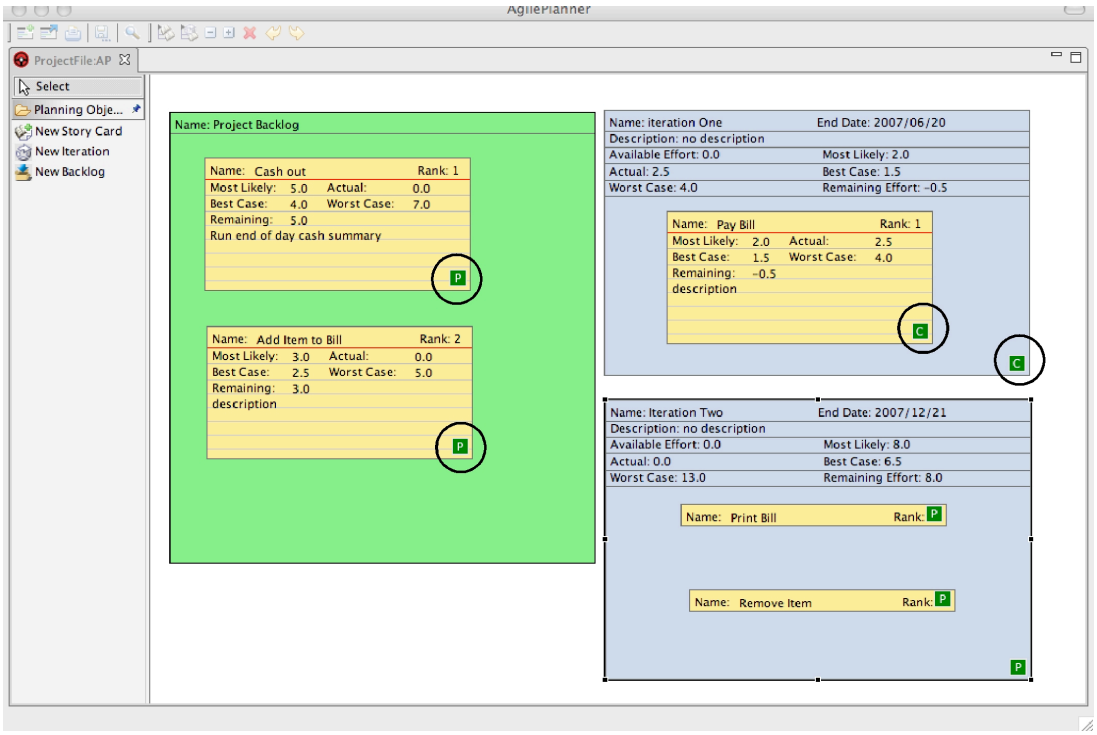


Figure 4.15: Iteration and story card status

4.2 Distributed Planning Features

DAP's primary focus is to support distributed teams conducting planning meetings. The features presented above can be seen in many existing planning tools and do not differ from them to any great extent. The innovative aspects of DAP come from its use in a distributed setting. Many of the existing tools [Reeses et.al. 2004, EBE 2007, Floranta 2007, Rally 2007, Danube 2007, XPlanner 2007] provide little to no support for updating and sharing information in near real-time while at the same time providing immediate feedback to the other team members using the tools. A majority of existing tools require the user to initiate the update for the project information. DAP takes great care in providing as much information to the team as possible when it comes to

interacting with planning objects, and when it comes to distributed teams this is no different.

To improve upon synchronous information sharing, DAP takes the approach that every change to the planning environment needs to be shared with every other client immediately. This approach means that every interaction is sent to every client that is connected to the project in near real-time. For example, as soon as a planning artefact is created on one client a message is sent to every other client indicating that a new object has been created. The message includes the information needed to replicate that object on the other clients. The same kind of approach occurs when an object is edited. Changing the name, location, etc. causes the other clients to be simultaneously updated in near real-time. This ensures that the planning environment is never out of sync. (see Chapter 5.3 for DAP performance)

4.2.1 Telepointers

Sharing artefacts in this way allows for the teams to see the state of the project plan in a way that simulates a collocated team. A large portion of the time spent planning is spent in discussion and during these discussions team members often refer to planning artefacts. When discussing in a collocated setting, it is easy to pickup or point to a story, however, this is much harder when you're in a distributed setting. Often you would have to describe the card by name, or some other distinguishing attribute. This is not a natural way of communicating during the planning process and yet pointing with your hand at your display is of no help to those that can't see your hand. One solution is providing remote mouse cursors in the workspace in order to support pointing and gesturing.

To support pointing to artefacts in the shared workspace DAP uses telepointers [Dyck et.al. 2004] to simulate pointing and gesturing. This lets the team members using DAP use their mouse to point to objects on their display and have the telepointer (Figure 4.16) mimic the users' movements on the subsequent clients. In cases where a large number of clients are interacting with the same project it can become quite difficult to distinguish between the various individuals telepointers. To help distinguish which telepointer is associated with which client by, the client's initials or name (Figure 4.17) are added to the telepointer.

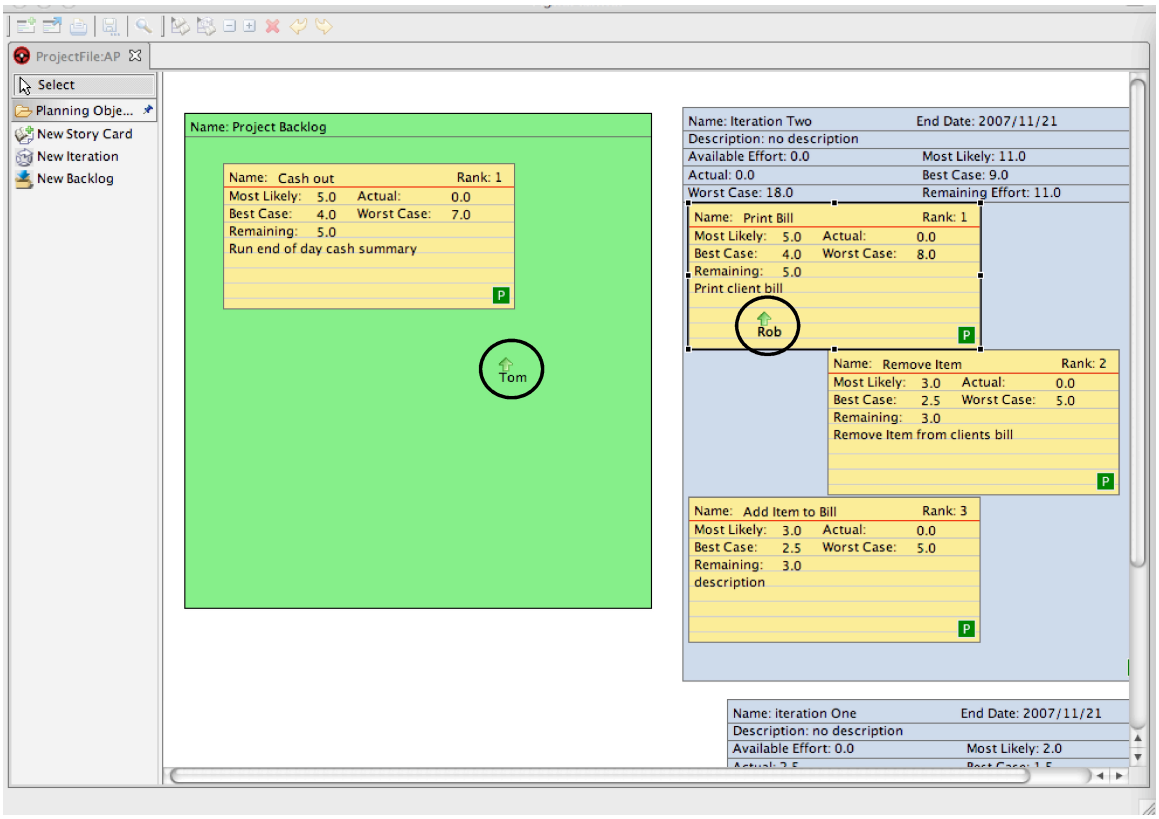


Figure 4.16: Telepointers in DAP

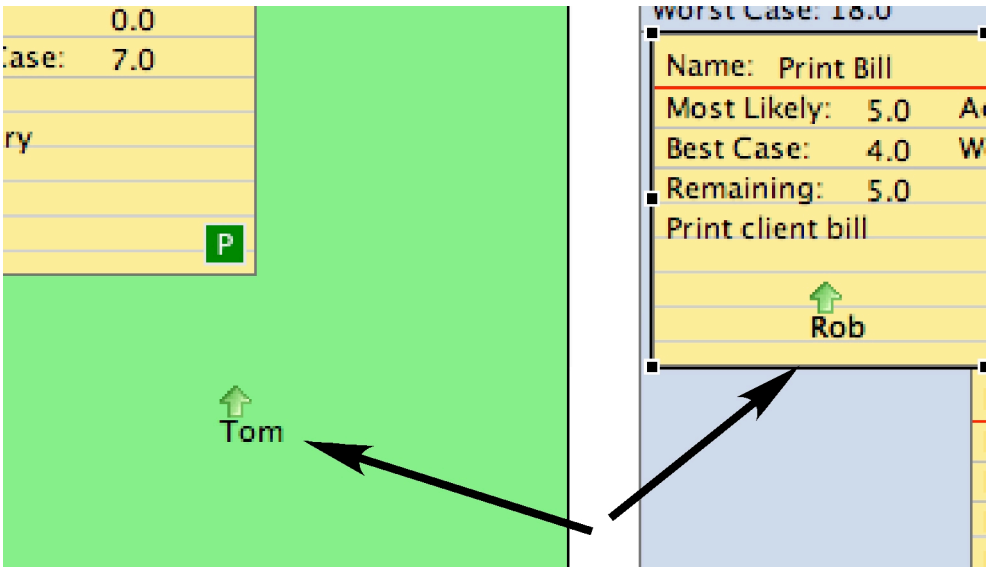


Figure 4.17: Telepointer ID's

4.2.2 Relaxed WYSIWIS

To ensure that everyone is able to see the interactions of others an extremely relaxed WYSIWIS implementation was used. As the number of artefacts in the shared workspace can be significant and screen sizes and resolutions vary the project workspace grows and shrinks as needed. This results in the need to allow scrolling in the various different directions. The scroll bars double as a very simple awareness mechanism so that when telepointers are outside a given user's view port, the scroll bars grow and shrink with telepointer movement.

Using the scroll bars as the only means of workspace awareness of interaction outside one's view of the project is extremely limited. However, it maximizes the visual planning surface and allows teams to make use of as much of the visual surface as possible.

4.2.3 Live text

Augmenting a telepointer's ability to provide awareness and live feedback to distributed team members in DAP is a feature that we call Live Text. This live text follows the same lines as telepointers in that it tries to provide distributed teams with as much information and feedback as they would have if they were collocated. As the name implies, live text displays changes to a given piece of text of planning artefacts as the text is changing (Figure 4.18).

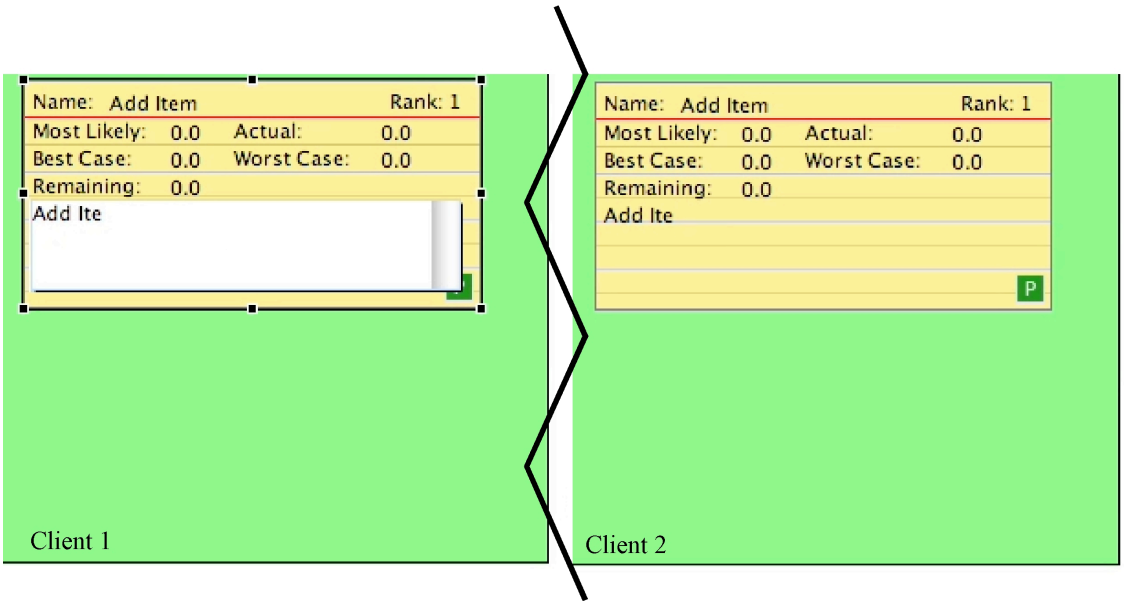


Figure 4.18: Live text

4.2.4 Project Sharing

Supporting distributed teams with DAP requires the project data to be shared with every client. In order to share the project data, DAP uses a client server model whereby the project information is stored on a dedicated server and each client connects to the server to interact with the project. Interacting with a shared project means that every

action that occurs on a given client is immediately sent to the server which in turn updates the project and transmits the update to the other connected clients. The server maintains a connection with each connected client to ensure that changes at either end are transmitted immediately.

The server, like existing planning tools, provides a central location to store project data for future access. This is important for planning as it allows for easy recreation of the planning state at a later time. This ensures that when planning information is changed during the course of an iteration, everyone has the updated changes when they connect to the project. This limits the risk that a given team member is using project data that is out of sync.

4.3 Implementation Details

4.3.1 Client Application

DAP is written in Java [Sun 2007] as an eclipse [Eclipse Foundation 2007: Eclipse] Rich Client Platform (RCP) [Eclipse Foundation 2007: Rich Client Platform] plug-in. This choice of implementation allowed for the use of existing frameworks to facilitate development of the user interface. These frameworks include draw2D [IBM 2005] and the Graphical Editing Framework (GEF) [Eclipse Foundation 2007: Graphical Editing Framework]. In using GEF it was necessary that DAP follow a Model-View-Controller (MVC) implementation (see GEF online documentation for more on MVC [Eclipse Foundation 2007: Graphical Editing Framework]). This approach allowed DAP to have a centralized model - a java object that contains the planning artefact information, which could be synchronized with the server with little effort. This model is responsible

for triggering user interface updates when a message is received from the server (Figure 4.19: step 6) and to send updated model information to the server when the user interface is changed (Figure 4.19: steps 1 & 2).

The communication layer for DAP is comprised of two dedicated threads to handle incoming and outgoing messages respectively. Sending messages to the server requires the communication layer to combine the model with the changes that are to be made to it and send this to the server. The server will process the incoming change (Figure 4.19: steps 3,4, & 5) and send the updated model to all the connected clients (Figure 4.19: step 6).

The second communication thread, which is constantly listening for incoming changes, receives a new message, decodes the message, separating the model from the additional information, and determines what model needs to be updated. The thread fires an incoming update event for the given model, which in turn updates the model and subsequently the user interface causing it to refresh. (See Figure 4.19: step 7)

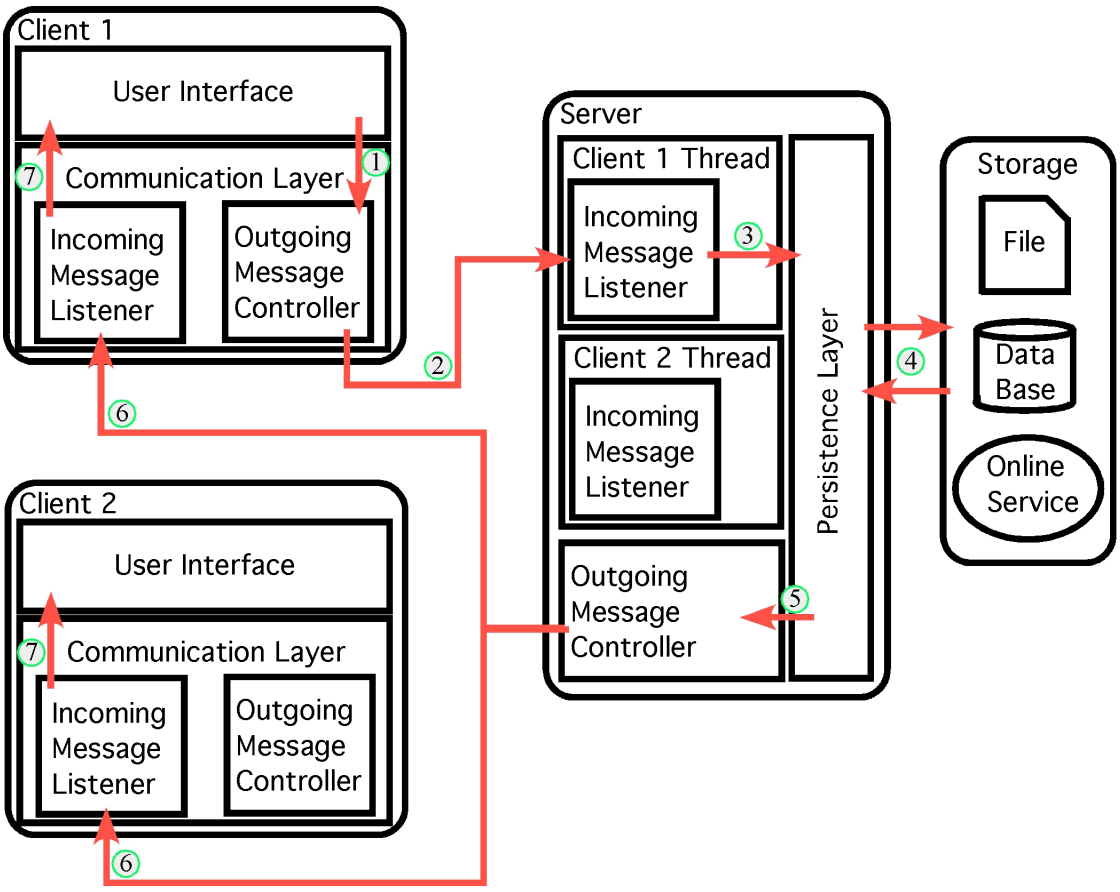


Figure 4.19: Client/Server Communication

4.3.2 Server Application

The server application (AP Server) is a simple java [Sun 2007] application. The server has two aspects that it is responsible for. First, it must handle incoming and outgoing model updates from the various different connected clients (Figure 4.19: Client threads). Second, it must persist the changes in some way (Figure 4.19: Persistence Layer & Storage).

The DAP Server, like the client, has a communication layer that handles the incoming and outgoing changes. It differs from the client in that it maintains a pair of threads for each connected client. This allows the DAP Server to control which client

receives what information and limit the amount of messages transmitted. This is important for two reasons: messages, like mouse movements and keys pressed, only need to be sent to remote clients and thus do not need to be returned to the sending client, unlike a model update, and second, a server can handle multiple projects and clients that are connected to a project. For example, project A does not need to process information pertaining to project B.

The persistence layer is responsible for maintaining a snapshot of the project after every change to a given model. Currently the DAP Server is using a file based system. However, the persistence layer was explicitly designed to be easily interchanged with a database system or any other custom implementation (Figure 4.19: storage). As of the time of writing the DAP server persistence layer has successfully been extended to connect to a commercial planning system.

4.4 Summary

This chapter presented the features available in DAP as well as some implementation details. DAP is an eclipse [Eclipse Foundation 2007: Eclipse] RCP [Eclipse Foundation 2007: Rich Client Platform] plug-in that makes use of the GEF [Eclipse Foundation 2007: Graphical Editing Framework] framework. Using a client server approach DAP provides users the ability to create and interact with story cards, organize them into iterations and estimate story effort while allowing team members to be distributed around the world. DAP uses Telepointers and Live Text to provide up-to-date information related to the discussions occurring during the planning. DAP attempts

to provide distributed teams with as all of the benefits of face-to-face planning without the need to be collocated.

Chapter Five: Tool Analysis

The goal of this work is to create a tool that supports distributed teams conducting agile planning meetings. In Chapters Two and Three, existing tools used for agile planning were presented as well as a list of requirements needed to support real-time distributed agile planning. This chapter looks at the existing set of tool (state of the art) in terms of meeting the requirements identified in Chapter Three. Also presented are the results of an evaluation of DAP against these requirements.

The results of my analysis are summarized in Table 5.1. Existing tools were evaluated against three possible outcomes: Full (F), Partial (P), or No (N) support for the requirements. Partial support was considered where there were possible degrees of fulfillment. Partial support for each of the requirements is defined as follows:

1. **Creating, editing and deleting planning objects.** Partial support for creating, editing and deleting indicates that the tool supports a subset of the three actions.
2. **Agile planning metrics management.** Partial support for metrics management indicates that support for some planning metrics is included; however at least one of the metrics, Best Case Estimate, Worst Case Estimate, Most Likely Estimate, Actual Effort, Remaining Effort, were not available.
3. **Planning multiple iterations.** Partial support for planning multiple iterations indicates that no explicit iteration could be defined; however, stories could still be grouped together to imply an iteration.

4. **Moving stories from one iteration to another.** Partial support for this requirement is related to the previous requirement. Partial support indicates that stories could be moved from one group to another, without the explicit existence of an iteration.
5. **Team and identity authentication.** There is no partial support. Tools either provided an authentication mechanism or they did not.
6. **Real-time exposure of the plan via the Internet.** There is no partial support. Tools either provided remote access to project data or they did not.
7. **Visual characteristics for different types of stories.** No partial support. Story Cards either had visual characteristics or did not.
8. **Integration with the development environment.** Partial support indicates that tools are capable of being used inside a integrated development environment (IDE) with limited functionality (an example would be an eclipse plug-in [Eclipse Foundation 2007:Eclipse]).
9. **Fluid transition between individual and collaborative work.** Partial support indicates that there is no explicit personal space in the tool however, the tool does not prevent users from switching to another application.
10. **Telepointers for pointing and gesturing.** There is no partial support. Planning tools either supported telepointers or they did not.
11. **Flexible information sharing (WYSIWIS).** Partial support indicates that tools support a relaxed implementation of WYSIWIS (Chapter 2.3.1) without the knowledge of knowing what other are viewing (their Viewport) [Gutwin et.al. 1998].

12. **Change notification.** Partial support applies to tools that depend on notification of changes after the changes have occurred or require users to be viewing a specific part of the workspace in order to receive the change notification.
13. **Radars for awareness information.** There is no partial support. Planning tools either provided radars for awareness or they did not.
14. **Flexible participant management.** There is no partial support. Tools either supported participants entering and leaving at any given time or they did not.
15. **Fluid subgroup formation and dissolution.** Partial support indicates support for casual subgroup formation without explicitly defining a subgroup or subgroup work area.
16. **Simultaneous interaction.** Partial support indicates simultaneous interaction by different instances of the tools is not fully supported in all areas of the respective tools.

In evaluating the state of the art against DAP we can see that for the majority of the cases DAP provides full or partial support for the requirements, and is deemed a success as it fulfils more of the requirements then any other means of distributed planning.

5.1 State of the Art

Existing tools used for planning in agile teams vary from a shared whiteboard or sharing spreadsheets to using customized online applications. This section looks at

existing agile planning tools and how these tools satisfy the requirements presented earlier.

Physical index cards are often used for planning in both collocated and distributed settings. In a distributed setting, paper cards are not ideal in terms of a planning medium. This is due to the fact that only team members at a single site can interact with the cards. Paper story cards are easy to create, edit, delete (destroy), and they can easily be moved from one iteration to another, allowing collocated teams to plan more than one iteration at a time. However, in a distributed setting they cannot be shared with team members unless the team ensures that all the story cards are duplicated and organized identically across all locations which results in additional effort on the part of the team. Thus none of the requirements is fulfilled for distributed teams using paper cards because the information is not shared with the other sites involved with the planning.

In looking at the groupware requirements, a limited number of these requirements is supported by existing tools. Personal and private spaces exist; however, there is no means of sharing hand gestures, notifying others of changes without explicitly telling them that you are changing the cards. Essentially, much of the awareness information is missing for the distributed team members.

Online tools such as [EBE 2007, Floranta 2007, Rally 2007, Danube 2007, VersionOne 2007, XPlanner 2007] improve upon paper index cards by providing the same information to all team members at all distributed locations. These tools generally provide similar features and will be grouped together. All these tools provide support for creating, editing and deleting story cards, planning multiple iterations and moving stories between them. These tools use a tabular presentation (Figure 5.1) of planning artefacts

making it easy to show planning metrics. Thus, for the most part, the online tools fulfill most of the agile specific requirements with the exception of integration with the development environment. Their limitations stem from not supporting most of the groupware requirements. The groupware requirements that they do support are change notification and flexible participant management. Some of the tools deliver email notification when planning information is changed and as the systems are accessed via the web team members can access the information at any given point, always receiving the latest data.

Criteria	DAP	CardMeeting	AgilePlanner	MASE	Rally	VersionOne	Xplanner	Scrumworks	GlueWiki	DotStories	XPSWiki	Moomba	Tukan
Creating/Editing/Deleting Artifacts	F	F	F	F	F	F	F	F	F	F	F	F	F
Planning Metrics Management	F	N	F	F	F	F	F	F	N	N	F	P	P
Planning Multiple Iterations	F	P	F	F	F	F	F	F	P	P	F	F	N
Moving Stories between Iterations	F	P	F	F	F	F	F	F	P	P	F	F	N
team & Identity Authentication	N	F	F	F	F	F	F	F	F	F	F	?	?
Real time exposure via Internet	F	F	F	F	F	F	F	F	F	F	F	F	F
Visual Characteristics for Story Cards	N	F	F	F	F	F	N	N	F	N	N	N	N
Integration with the development Env.	F	N	N	N	N	N	N	N	N	N	P	N	N
Fluid Transition between personal/ private space	P	P	F	P	P	P	P	P	P	P	P	P	P
Telepointers	F	N	N	N	N	N	N	N	N	N	N	N	N
Flexible Information Shairing (WYSIWIS)	P	P	P	N	N	N	N	N	N	?	N	F	F
Change Notification	P	P	N	P	P	P	?	P	N	?	?	F	F
Radars for Awareness	N	N	N	N	N	N	N	N	N	N	N	N	N
Flexible Participant Management	F	F	F	F	F	F	F	F	F	F	F	F	F
Support for Subgroups	P	P	P	N	N	N	N	N	P	P	?	?	?
Simultneous interaction	F	F	P	N	N	N	N	N	N	?	?	?	P

?=Not enough data avaliable

Table 5.1: Tool Comparison

Of particular interest is CardMeeting [CardMeeting 2007]. This online tool does not follow the typical approach taken by other planning tools. CardMeeting, like DAP, is modeled after the physical story card planning. In doing so it fulfills many of the requirements both from the agile and groupware sets, such as visual characteristics, for different cards and simultaneous interaction for the different clients. The agile-related limitations stem from the strict following of the index card metaphor, there is no way of keeping track of planning metrics, and iterations are not explicitly defined thus a summary of the information is not available.

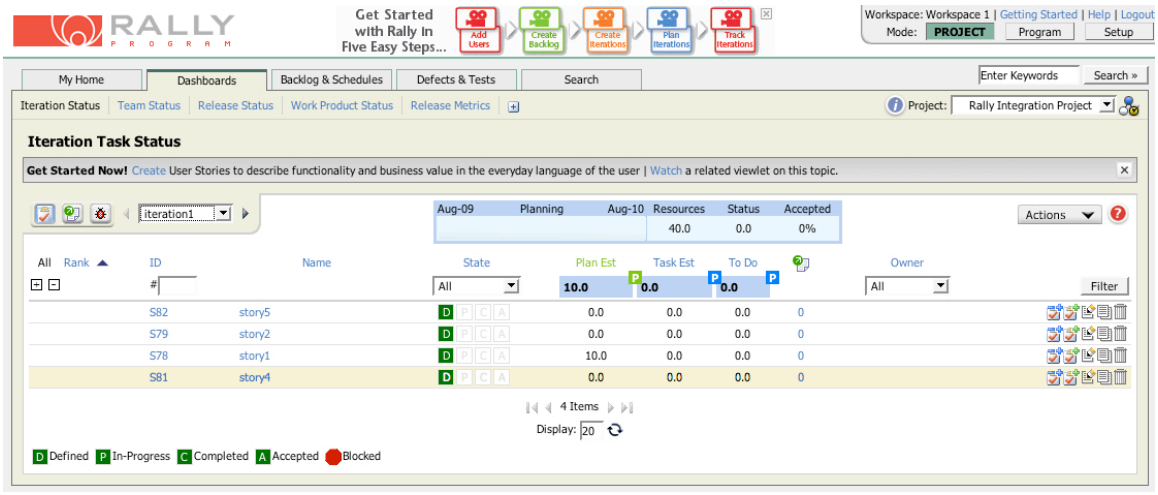


Figure 5.1: Tabular presentation of data [Rally 2007]

Requirements from the groupware community are better supported than in any of the existing tools. Supported requirements include supporting simultaneous interaction with visual representation of the cards and using a relaxed WYSIWIS interface. The limiting factor in CardMeeting is its inability to support gesturing or pointing in the workspace, and, like all other tools, it is limited in its support of simultaneous interaction.

Simultaneous interaction is supported for single distributed team members. Teams with more than one team member collocated are limited by the computer's inability to support simultaneous users.

Performance in terms of response times can vary significantly from interaction to interaction (Table 5.2). An example is when testing CardMeeting with two clients in the same computer lab, the time to create a card in the planning workspace varied between 1.5 to 3.75 seconds. Moving story cards had less of a delay with times of 1.75 to 2.0 seconds before the movement could be seen on the other client. The most noticeable delay came from editing card content. Changes to card text were not visible to other clients until the changes to the individual card were completed and the user clicked on the workspace (deselecting the card). Once the card was deselected changes took between 2.0 to 2.5 seconds to be displayed on the other clients' workspace.

Create Story Card	1.5 - 3.75 sec.
Move Story Card	1.75 - 2.0 sec.
Edit Text (Delay after deselect)	2.0 - 2.5 sec.
Delete Story Card	1.5 sec.
Reload Project Data	10 - 17 sec.

Table 5.2: CardMeeting Performance Times

5.2 DAP supported Criteria

With respect to supporting the requirements presented in Chapter Three, DAP fully or partially satisfies thirteen of the sixteen requirements (Table 5.1). Of the

supported requirements, DAP better supports requirements stemming from the agile planning subset rather than the groupware set.

The DAP planning environment and digital representations of planning artefacts provides users with the ability to create, edit and delete digital representations of iterations, a product backlog, and story cards. In supporting the create, edit, and delete features with digital representations of planning artefacts, DAP fully satisfies both the need to support creation, editing, and deletion of planning artefacts and the need to provide shared access to digital planning objects.

By presenting the artefacts as two-dimensional objects in a shared workspace users are able to interact with the planning artefacts in a way that better resembles interacting with paper story cards. The use of drag-and-drop or point-and-click makes creating, moving or organizing story cards much easier. This means of direct interaction with the story cards specifically makes moving cards from one iteration to another more fluid thus allowing teams to plan multiple iterations while keeping the focus on the discussion and not on interacting with the tool.

DAP's support for flexible sizes and of drag-and-drop for moving artefacts provides teams with the ability to plan multiple iterations at a given time and move story cards from one iteration to another with ease. Iterations are easily moved around the workspace, along with their contained story cards, to allow teams to add or change focus to another iteration or backlog.

Story cards provide editable fields for storing estimated and actual effort. DAP supports collecting the most likely effort and any combination of: best case, worst case, actual, and remaining effort. The planning metrics are summarized in the header of the

iteration artefact and are updated when contained story card values are changed or when a story card is added or removed from the iteration. Story card priority is also supported and is dynamically updated when a story card is added to the iteration or when the vertical positioning of the card is changed.

DAP's planning environment is a shared workspace with all other DAP clients connected to the same project. The connection between clients is provided via a network connection. This exposes the project plan to any client with a network connection, provided that the DAP server is connected to the Internet. The project plan is updated in near real-time (see section 5.3) due to the way that DAP's server maintains a direct connection with each client. Changes to the project plan are pushed from the client to the server and the server subsequently pushes the changes out to all other connected clients.

In order to better support both developers and non-developers alike, DAP is integrated with the eclipse [Eclipse Foundation 2007: Eclipse] development environment in addition to being used as a standalone application. DAP is built as an eclipse plug-in and, as such, team members have the option to use the tool as part of the eclipse environment or not.

In order to better support collaboration among distributed team members, DAP supports telepointers in its shared workspace. The telepointers allow team members to use their mouse to point to artefacts or regions in the workspace and have this awareness information be visible on all other clients. Telepointers include the user's name or their initials thus allowing others to identify their interactions in the shared workspace. The telepointers support more fluid communication between sites as individuals can communicate more naturally.

Managing team members entering or leaving the planning meeting is supported automatically in DAP. Participants can enter or leave the collaborative setting at their convenience. As new members connect to the project they are immediately sent an updated project plan. Telepointers are automatically added to the other clients once the new members starts moving their mouse in the shared workspace. Upon exit, the client disconnects from the server and changes are no longer sent out via that connection. The leaving users' telepointer is subsequently removed from the remaining clients' screens indicating that they are no longer viewing the project plan.

Sharing plan information is partially supported in DAP with a relaxed WYSIWIS shared workspace. Changes to the project plan are shared with all connected clients and the changes are reflected in the workspace; however, screen real estate limits how much of the workspace is visible. In order to provide the most flexibility for each connected team member, each client's view of the project can be changed without impacting the view of other clients. Information sharing is limited due to the fact that the spatial awareness of where others are located is only available through visible telepointer locations or changes in the size of the visible workspace's scrollbars.

The relaxed WYSIWIS workspace does allow for easy formation and dissolution of subgroups similar to what could be observed in a collocated team sitting around a table. In DAP, team members can easily join or leave a subgroup by scrolling to its location in the workspace. Subgroup formation is only partially supported here due to the fact that there is no means to explicitly define a subgroup [Stefik et.al. 1987]. There is also no way for the subgroup to keep apprised of what is changing in parts of the workspace that is outside of a given client's view (change notification).

Notifying others of change is only partially supported in DAP. The live text feature provides notification and feedback to connected users that the content of an artefact is changing. The notification that text is changing is limited in that an individual needs to be connected to the project plan and they must have the artefact in the visible workspace. There is no mechanism, currently in place, that notifies offline users of changes nor is there an indicator for users viewing a different portion of the project plan.

Interacting with artefacts in the shared workspace is an essential part of DAP. DAP fully supports simultaneous interactions in the case where the interactions are occurring on different clients. The same cannot be said of concurrent interactions on the same client. The possibility of having multiple team members at a location is highly likely and in collocated planning it's often the case that more than one team member will be interacting with the story cards. Like all other planning tools, this is not supported in DAP. In order for two or more team members to interact with the planning environment, each must have their own instance of DAP running on a personal computer.

5.3 Real-Time Performance

Response time in Distributed AgilePlanner is in near real-time. Actions in DAP were timed in both a laboratory setting (university lab) and in an inter cross-continent setting (Calgary, Canada – Lancaster, United Kingdom). For both time trials the DAP server was located in at the University of Calgary. Table 5.3 summarizes the times for the various different actions.

The majority of actions in DAP took two seconds or less in both settings, with the results from the lab setting being considerably faster. For example, creating a story card

took 0.75 seconds before it appeared on the client in the UK and less than 0.5 seconds when tested in the lab. Moving an empty iteration took between 0.25 seconds to 1.5 seconds and when twelve story cards were added to the iteration the times increased to 1.75 seconds and 2.0 seconds for the lab and inter cross-continent trials respectively.

	Calgary, Canada – Lancaster, United Kingdom	Calgary, Canada – University Lab
Create Story Card	0.75 sec	0.5 sec
Create Iteration	1.5 sec	0.5 sec
Move Story Card	1.2 sec	0.5 sec
Move Empty Iteration	1.5 sec	0.25 sec
Move Iteration with 12 Story Cards	2 sec	1.75 sec
Move mouse	1.5 sec	0.25 sec
Edit Text – Short (Story Card) Less than 10 characters	1.5 sec	0.25 sec
Edit Text – Long (Story Card) over 75 Characters	2 sec	0.25 sec
Delete Story Card	1.5 sec	1.0 sec
Reload Project Data	5 sec	2.7 sec

Table 5.3: DAP Response times

The results from the lab experiment were relatively instantaneous; however, we can see (Table 5.3) that there is a delay introduced when collaborating over longer distances. The delay introduced is considerably short and in many cases went unnoticed during testing resulting in the test being repeated numerous times.

Timing of the text editing was timed from the start of typing at one location until the end of typing at the remote location. Two time tests were conducted to show any delays introduced by larger amounts of text. We can see that the difference in time did not change in the lab environment; however, when collaborating with a client in the UK a difference of about 0.5 seconds was observed between short text updates and longer text updates.

Timing of all the actions was conducted with a stopwatch with all values rounded up to the nearest quarter of a second. When an action was triggered the individual executing the action would say “now” to indicate that the timer should be started and the other client would say: “done”, when they saw the action occur on their display. This approach to timing the actions does introduce a slight overhead in the timing, resulting in an increased timed value. With this overhead taken into account the resulting times for the various actions in DAP remain quick.

Performance in DAP is quick and results in team member interactions with the planning environment being shared in near real-time with connected clients. We can see from our time trials that geographic distance between clients does have a small impact on the performance; however the response time remains under two seconds for the majority of actions.

5.4 Limitations of DAP

Despite all the supported and partially supported requirements, DAP falls short in a handful of areas. Supporting different types of story cards is one area where DAP falls

short. Newly created story cards are identical in look and colour[1]. Supporting different types of story cards was not implemented because observations showed that when dealing with physical cards, it was not always the case that the team had different coloured cards to denote different types of stories and in cases where multiple colours were available. keywords such as “bug” preceded the story name. As such, supporting this requirement was postponed.

Authentication for teams was also postponed as the ability to prevent unauthorized users changing the project plan was deemed to be a low priority and would not affect the evaluation of DAP to support distributed teams. It is important to note that industrial tools must support team authentication; however for research prototypes authentication mechanisms are not as important.

The high emphasis on collaboration in DAP lead to the sacrifice of individual space in the workspace. DAP’s shared workspace supports collaboration and we chose not to support individual workspaces. As team members can easily switch to an individual workspace existing outside of the DAP environment (either by changing to another non-shared application or to a physical workspace) the implementation of an individual workspace was omitted from the planning environment and as such partially fulfils this requirement.

---

[1] Continued development on DAP has provided story cards with a colour option.

5.5 Summary

We have seen that the existing state of agile planning tools satisfies many of the agile-related requirements but falls short when it comes to supporting the groupware requirements. Tools are improving and CardMeeting is one tool that has satisfied some of the groupware requirements on top of many of the agile requirements.

DAP is able to satisfy thirteen of the sixteen requirements presented in Chapter Three. Although it does not satisfy all of the requirements, it's moving in the right direction in order to better support distributed agile teams.

Chapter Six: Qualitative Evaluation

This research looks at building a tool to improve the distributed planning process for agile teams. One of the goals for this research is to build a planning tool that better supports the planning process in an intuitive and familiar way. In the previous chapter, Distributed AgilePlanner (DAP) was evaluated against the criteria identified from previous work in the areas of both agile methods and groupware. In this chapter, we look at how the tool supports planning in a distributed environment. A qualitative evaluation was conducted. The evaluation is comprised of two preliminary case studies and two preliminary user studies. Following the case studies and user studies, semi-structured interviews were conducted with the study participants.

6.1 Objectives

The qualitative evaluation of DAP intends to determine if the tool supports distributed planning better than traditional paper-based planning. In addition, the evaluation aims to determine whether a virtual shared workspace supports more natural interactions and a more intuitive planning environment for all parties involved.

Besides looking at how DAP supports agile planning, the evaluation also aims at finding shortcomings in the tool. Research done by Grudin et.al. [Grudin 1994] pointed out that evaluating groupware applications, in terms of usability, is affected by the group dynamics and the backgrounds of the individual participants. In an attempt to overcome this concern, five groups were observed amounting to twenty-four participants.

6.2 Participants & Context

The evaluation of DAP was conducted over a six month period in 2007. During this timeframe, five teams were observed as they conducted their planning meetings. The teams participating in the two case studies were developing software for a client. The teams involved in the user studies were graduate students volunteering their time to participate in the study. All the teams were asked to participate in interviews after the planning meetings (see Figure 6.1).

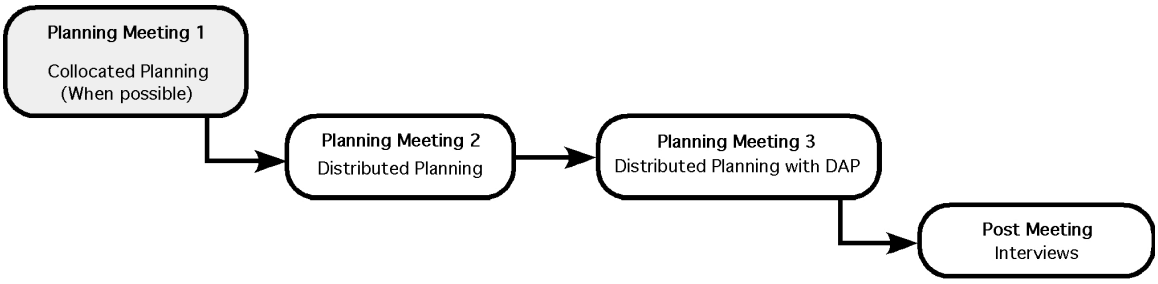


Figure 6.1: Qualitative Evaluation Timeline

6.2.1 Case study teams

The first team was the Distributed AgilePlanner development team. This team involved a customer based in the United States and the developers based in Canada. The customer was a senior development manager of a medium sized software firm with over 10 years of experience in the agile community. The development team consisted of graduate students and internship students located in Calgary, with varying experience with agile methods. This team had been working together for one and a half years with new developers joining the team and some leaving the team during the software development. As this team was the development team for Distributed AgilePlanner, and

had been using the tool prior to the evaluation, the use of the tool was continued for all three planning sessions with a conference phone for supporting audio communication. (Table 6.1)

The second team included graduate students taking an agile methods course. This course started in January 2007 and the team was observed throughout the planning meetings of the course project. Participants in this case study had varying experiences with agile methods and were being instructed throughout the course. The customer for this project was a professor at the university who was not the course instructor. The project was centered on developing a plug-in for a development environment for which the customer had some knowledge. The iterations for the project were approximately four weeks in length with teams working part-time. This team was entirely based in Calgary and was observed planning in three different scenarios; first was collocated with the customers and developers in the same room. In the second and third planning meetings the customer was located on a different floor than the developers. This was to simulate a distributed working environment. To make a comparison, the team used paper index cards with a conference phone for the second meeting and used DAP with a conference phone for the third meeting. (Table 6.1)

Participant	*Agile Experience*	*Group*	*Location*	*Observed*	*Feedback*	*Project Description*
P1	*10+ yr.*	*DAP*	Boulder	*N*	*N*	DAP Dev.
P2	1-5 yr.	*DAP*	Calgary	Y	Y	DAP Dev.
P3	1-5 yr.	*DAP*	Calgary	Y	Y	DAP Dev.
P4	1-5 yr.	*DAP*	Calgary	Y	Y	DAP Dev.
P5	1-5 yr.	Grad Course	Calgary	Y	Y	Eclipse Plug-in
P6	1-5 yr.	Grad Course	Calgary	Y	Y	Eclipse Plug-in
P7	Unknown	Grad Course	Calgary	Y	N	Eclipse Plug-in
P8	None	Grad Course	Calgary	N	Y	Eclipse Plug-in
P9	Unknown	Grad Course	Calgary	Y	N	Eclipse Plug-in
P10	Unknown	Grad Course	Calgary	Y	N	Eclipse Plug-in

Table 6.1: Case Study Summary

6.2.2 User study teams

The user studies were conducted with graduate student volunteers from Canada and the United Kingdom. Three teams were created. Each team had three developers based in Calgary and three customers, one based in Victoria (Can) and two in Lancaster (UK). The teams had varying experience levels with agile methods and story-based planning with some of the participants having no prior experience. The participants based in the UK were business management students and had the least experience with agile methods. The participants from Victoria were computer science students with some to no agile experience. Participants in Calgary, who were acting as developers, had at least one person per team with over one year of agile experience and generally were the more experienced participants. (Table 6.2)

Participant	*Agile Experience*	*Group*	*Location*	*Observed*	*Feedback*
P11	5-10 yr.	UK-One	Calgary	Y	Y
P12	1-5 yr.	UK-One	Calgary	Y	Y
P13	None	UK-One	Calgary	Y	Y
P14	None	UK-One	Lancaster	Y	Y
P15	None	UK-One	Lancaster	Y	Y
P16	None	UK-One	Lancaster	Y	Y
P17	1-5 yr.	UK-Two	Calgary	Y	Y
P18	Unknown	UK-Two	Calgary	Y	N
P19	1-5 yr.	UK-Two	Calgary	Y	Y
P20	None	UK-Two	Lancaster	Y	Y
P21	None	UK-Two	Lancaster	Y	Y
P22	None	UK-Two	Lancaster	Y	Y
P23	5-10 yr.	Victoria	Calgary	Y	Y
P24	1-5 yr.	Victoria	Calgary	Y	Y
P25	None	Victoria	Calgary	Y	Y
P26	1-5 yr.	Victoria	Victoria	N	Y
P27	None	Victoria	Victoria	N	Y

Table 6.2: User Study Summary

All three groups were given the same high-level project description. The teams were to build a point-of-sale software application. The description was left vague (Appendix B) to allow for the teams to come up with their own stories and conduct the planning meeting in a way that best fit their perception. Teams were asked to conduct a story-based planning meeting with estimates and priorities for the stories.

The user studies involved the teams conducting two planning meetings for the project. The meetings were allotted one hour for each planning meeting, after which the participants were asked to wrap up the meetings and stop. In the first meeting, the teams only used paper index cards and a conference telephone. In the second meeting, the DAP tool was applied along with the telephone. The user study teams differed from the case study meetings as there was no development involved and the meeting length was fixed due to scheduling restrictions on the part of the participants.

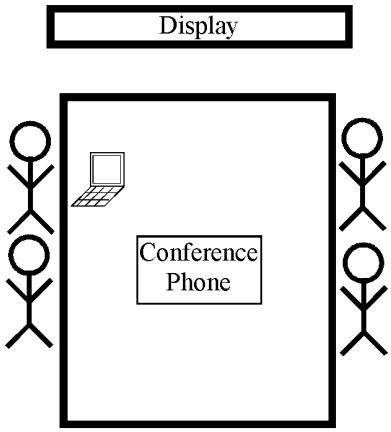
6.3 Data Collection & Participant Environment

Data was collected from the teams through observations of the planning meeting and the interviews. Teams were observed as they conducted the planning meeting by means of the researcher taking notes during the planning. In addition, the teams were video taped during the planning meetings. Video was only available at the Calgary and Lancaster locations as some locations had trouble arranging setup of a video system. Following each planning meeting the team members were asked to schedule an interview with the researcher in order to provide some feedback. Some of the participants did not make themselves available for interviews.

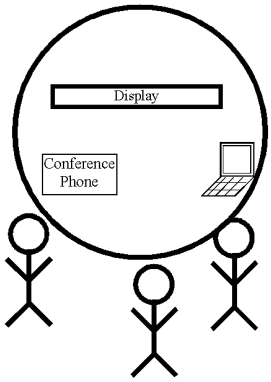
Team seating arrangements varied based on the teams' preference. Teams were given the freedom to arrange themselves as they saw fit. However, due to the sensitivity of the conference phone used, this typically resulted in teams sitting as close to the phone as possible.

The development teams located in Calgary used the same conference room with a conference phone located in the center of the conference table (Figure 6.2:A). This room also had a large rear projected screen to allow for teams to have easy viewing of the DAP workspace. When using DAP, one laptop was provided for the developer teams to use. Customer groups were a little more varied in their setup.

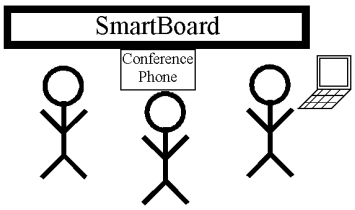
A)Calgary Conference Room



B) Calgary Conference Room



C) Lancaster



D) Victoria

Figure 6.2: Room layout

Customers in Victoria were conducting their planning meeting in an open laboratory space. For the first meeting they used a participant's desk for the planning. In the second meeting, the team moved to another part of the lab where a Smart Board [Smart Technologies 2007] was used to interact with DAP (Figure 6.2:D). For the teams located in Lancaster, UK, the teams conducted their planning meeting by sitting around a table in a professor's office. In the second meeting, the teams used the computer in the professor's office in order to interact with DAP (Figure 6.2:C). Verbal communication

for all three user study teams was provided by a telephone with speakerphone capabilities.

The customer, in the graduate course case study, was located in an office and used a standard office telephone to communicate with the developers. For both distributed planning meetings the customer was sitting at his desk, using their personal laptop for the meetings involving DAP.

6.4 Evaluation Criteria

Evaluation of DAP is based on the feedback provided by the participants over the course of the study and on the observations as well as a review of the video data collected. To determine the impact of DAP on supporting agile planning as well as the usability, the following questions were asked when reviewing the data:

- How did users perceive the learning curve?
- How intuitive did users perceive the interaction with the tool?
- What was the perceived ease of creating and interacting with the planning artefacts?
- How did teams feel about the tool's impact on the productivity of the planning?
- How was verbal communication affected by the introduction of the tool?
- How well did the groups interact together?

6.5 Observations & Feedback

Feedback and observations were positive towards the use of Distributed AgilePlanner. From the observations, the researcher (the author) saw the excitement and

a desire to use the tool. The feedback from the participants suggests that the tool did make a difference compared to traditional paper planning. One important note regarding this research is that all observations, interviews and tool introductions were conducted by the author.

6.5.1 Case Study Observations

6.5.1.1 Case Study One: DAP Development Team

In the first case study, the DAP development team used the tool for three planning meetings. The team's interaction with the planning tool was dominated by the developers. In all three meetings one developer and the customer were the only individuals to interact with the tool during the planning meeting. Meetings generally were dominated by verbal communication on the part of the customer, project manager, and senior developer. Interaction with the planning tool was minimal, as fourteen cards were created in the first planning meetings but only four in the second and third meetings.

During the three planning meetings, interactions with the cards were between the customer and one of the developers. Cards were generally created by the developer with only a hand-full of cards (over the course of the three meetings) being created by the customer. Cards were created by selecting the new story card button and clicking on an unoccupied part of the workspace. The backlog was not often used for creating cards, as it was the current iteration where new cards were created. Interactions with cards appeared effortless on the part of the developer and customer. During the planning meeting when the customer and developers were interacting with DAP, the customer commented: " …this is exciting. This is great!" - P1, referring to being able to interact

with artefacts and seeing the interactions from the developer's side. Interactions with DAP still had developers and the customer taking turns interacting with the planning artefacts; however, pointing and moving their respective mice around the screen occurred simultaneously.

Telepointers were a significant point of excitement on the part of the team. One participant was excited because of their ability to make the conversation more natural. They commented that: "It's the first time that I really saw that it was useful, because when [he] said look here!"- P2. Telepointers also allowed for the team to refer to story cards by pointing and using casual referencing (eg: "… this one here…"- P1). This made the conversations more casual and relaxed as it's similar to how collocated team reference cards.

6.5.1.2 Case Study Two: Grad Students

The second case study team participated in two different types of planning meetings: collocated and distributed. When comparing the collocated to the distributed planning everyone appeared to be more comfortable and engaged in the collocated planning. Eye contact with the customer and body language were most noticeably different. During the collocated meeting team members kept eye contact with the customer and kept their body facing him as well. During the first distributed meeting, as the customer was not present, team members eyes would wonder around the room and didn't appear to focus on anything in particular. In addition, they were noticeably more fidgety in their seats and appeared less interested in the conversation.

When comparing the two distributed planning sessions the team's perceived interest in the meeting was noticeably different. During the second distributed meeting, where teams used DAP, the team appeared to be more engaged in the meeting. The team's body language and eye contact were directed at the display where the story cards were being displayed. When compared to the first distributed planning meeting, only the project manager and one developer were conversing with the client and appeared to be interested in the meeting. The remainder of the team displayed various degrees of disinterest as they continuously looked around the conference room and fidgeted in their seats. These signs of disinterest were not seen during the second distributed planning meeting as the team members were focused on the display showing the story cards. The first distributed planning session also experienced a number of communication issues where the team members had to repeat themselves because the customer could not hear them or the question/comment was not explicitly directed to the customer.

Artefact interaction was limited to the development team, as they were responsible for the story cards. They were in possession of the story cards from the previous meeting and were responsible for creating new cards. While going over the cards, from the previous meeting, they read each card out loud in order for the customer to understand what card they were talking about. Card creation was primarily done by the project manager with additional cards being created by one of the developers. After the creation and estimation process, the customer began to ask questions specific to individual story cards. When trying to refer to a card he was unsure about he stated " I can't obviously point to [the card] but…." - P6. This led to a breakdown in the communication as the discussions would continuously stop and start to allow for reading

back of the story card and ensuring that the story description was correct as per the customers understanding. This resulted in the project manager and customer often loosing track of their thoughts and resulted in silence for the entire group. To ensure that both developers and customer had consistent story card data, the customer, at the end of the meeting, asked the developers to read back the cards. The project manager responded to this request by stating: "I'll type them up…." - P7. It was only after one of the developers pushed the project manager to read the card back, that the customers' request was fulfilled. However, this still did not synchronize the custome's and developer's visions of the cards. The development team had created six cards for the up and coming iteration, yet when the customer was asked following the meeting how many cards were in the iteration his response was four.

In the third meeting, the use of DAP helped to improve the team's ability to stay in-sync in terms of the stories created and included in a given iteration. Teams were given a thirty-second introduction of DAP, by the author, prior to using the tool. This introduction involved highlighting where the create story card button was located and a verbal note on how to move cards around. This limited introduction did not appear to hinder interactions with the tool. Both the customer and the development team interacting with DAP were able to create cards and organize them with little effort. One issue that did occur was when the project manager was trying to create a card on top of another. This is not supported in DAP and it took the project manager two attempts to realize this issue.

Resizing of cards was a common occurrence throughout the entire meeting. Screen real-estate was at a premium and the number of cards caused the team to compact

cards that they were no longer talking about (Figure 6.3). Teams only made use of the screen space visible and did not appear to know that the workspace was larger than the visible area.

The visual artefacts on the screen and the live text feature seamed to make the meeting's conversation more fluid when compared to the previous meeting's conversations. For example, the project manager and the customer had a conversation on the description of a story card. During the conversation, the project manager typed their understanding of the story card and asked if the description was correct. The customer quickly responded: "Ya, that makes sense"-P6. There was no reading of the card's description out loud or referencing the card explicitly. The conversation appeared fluid and was similar to the conversations during the first collocated meeting.

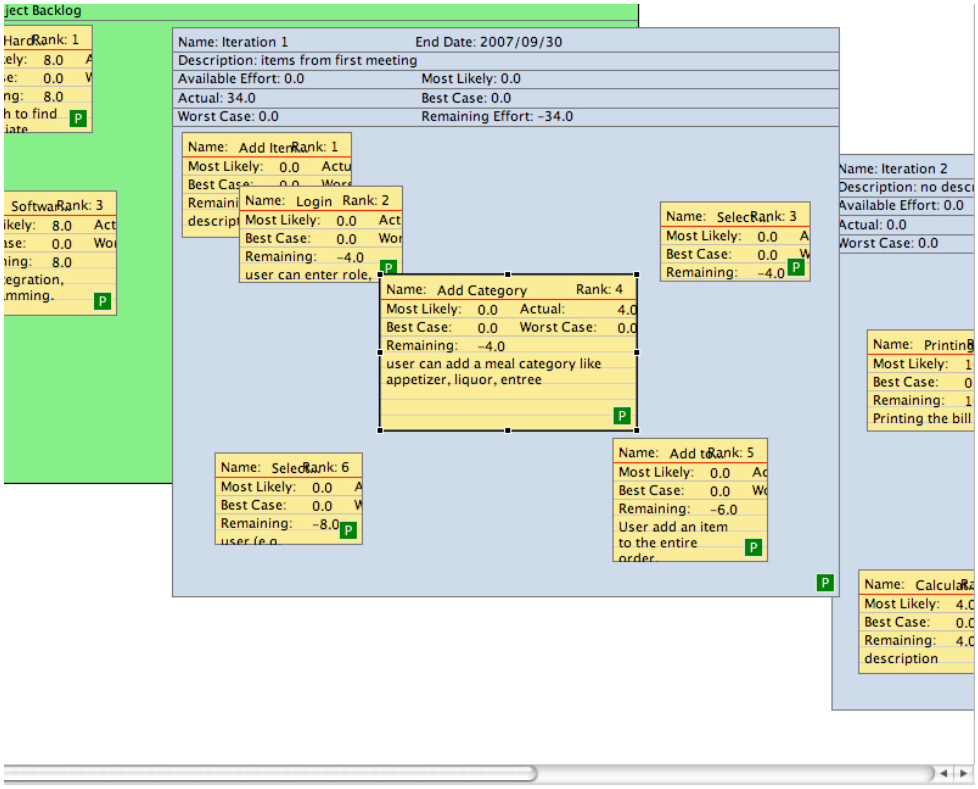


Figure 6.3: Resized card after discussion

During the planning meeting the project manager and one of the developers took turns interacting with DAP. This transfer of control between the two individuals was fluid; however, none of the other team members made any attempt to use the tool or participate in the planning meeting, with the exception of estimate creation. The customer's direct interaction with the workspace was evident by the movements of his mouse. This was most prominent during the ranking phase. The developers could see the re-ordering of the cards and the customer could use the mouse to point to cards of interest or to pose a question regarding a specific card.

One interesting occurrence that didn't occur during the first case study was assignment of tasks to individuals. The developer team used the description box to indicate who was responsible for a given card. In addition, two types of cards were created throughout the meetings: features and bug fixes. During the third planning meeting the bug fix cards were well distinguished from the other cards by means of the key word "Bug Fix" before the card name (Figure 6.4).

Overall, the team was a little hesitant following the use of the planning tool and requested a paper copy of the story cards. This can be in part, caused by technical difficulties encountered during the planning process where DAP experienced stability issues and required numerous restarts. The data was preserved; however, it created significant frustration among the entire team. The issues that caused the technical difficulties were resolved shortly after the planning meeting.

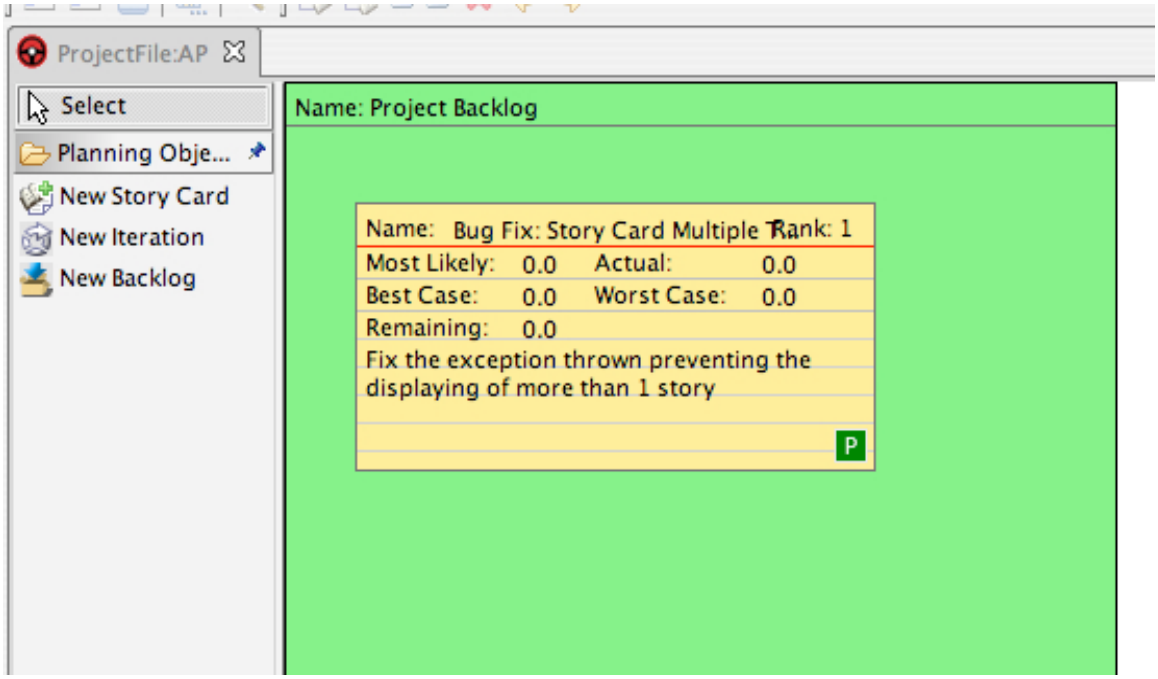


Figure 6.4: Use of "Bug Fix" to identify types of cards

6.5.2 User study Observations

The three user studies all experienced similarities when it came to interacting with the planning artefacts and communicating with each other. Each group, upon initial contact, had the customers providing an overview of the project followed by a brainstorming session. The length of the brainstorming sessions varied for each team. Communication was an issue for the UK groups during the first set of meetings.

In UK-One, for instance, one customer team was very excited about one feature that they described. At the same time, a developer was unsure about one thing mentioned and was attempting to interrupt and ask a question. However, over three minutes the developer attempted to interrupt ten times to no avail. This let to a significant amount of frustration on the part of the development team and the researcher saw a drastic change in

the body language and tone towards the customer group. In the developers' attempt to interrupt they repositioned their body numerous times before standing and positioning themselves directly over the speakerphone. The tone of voice was much firmer and commanding. This frustration increased throughout the remainder of the meeting. The development team was more assertive in asking questions and made a significant attempt to take the conversation control away from the customer.

In the UK-Two the communication issues coming from the development side were more subtle and were easily resolved. The issue experienced was more closely related to the creation of story cards and the terminology used therein. Traditionally, story cards are created by the customer [Beck 2000] and, as such, typically are written in terms that the customer understands. However, in all the distributed user studies, it was the developers who created story cards. The terminology the developers used on the story cards could not be understood by the customer. What was of concern was the fact that the development team's Project Manager, who controlled the conversation from the development side, was not making a significant effort to explain the terminology. Instead, the Project Manager simply repeated the content on the cards without giving a more detailed explanation. In the end, it was actually one developer who interrupted the Project Manager and explained the description to solve the issues.

The similarities among the three case studies extend to the way the story cards, once created, were organized. All three teams used columns (Figure 6.5) to organize the story cards. The number of columns was dependent on the number of story cards; however, there were at least two columns present for each team. The column organization carried over into discussions of story priorities with teams organizing cards with the

highest priority at the center of the table and working out to one of the sides as the priority decreased. Columns were used to differentiate between stories that were included in the iteration and those that were not. It's important to note that UK-One did not prioritize cards during the first planning meeting.

During the second set of planning meetings the similarities between the teams' interactions diminished slightly. Of particular interest, was the amount of interaction with the tool from the customer's perspective. Both UK groups' interaction with DAP was limited and appeared to be hesitant. The Victoria group on the other hand was excited and eager to use the tool and for a significant portion of the meeting was exclusively interacting with the tool while the developers participated through the discussions only.

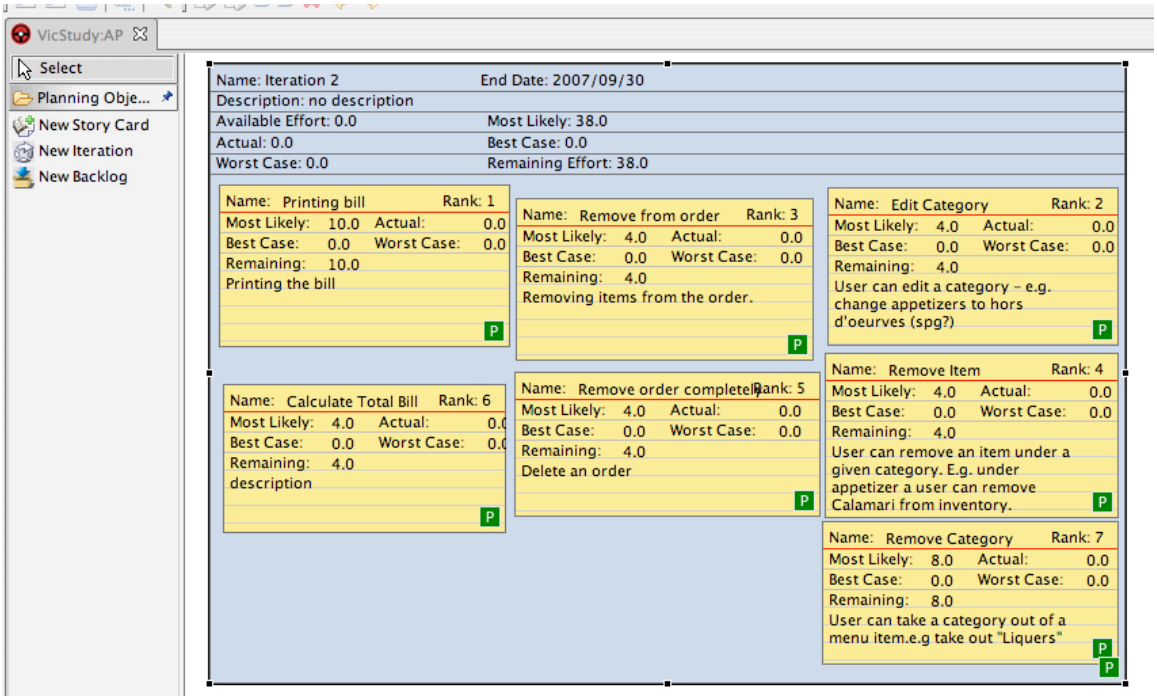


Figure 6.5: Use of columns to organize cards

When creating cards, in DAP, two of the three groups created cards by clicking at the location and dragging the card to the desired size. While the third group used a point and click approach. Interestingly teams were not shown the click and drag approach to creating cards when introduced to the tool and appeared to discover this as they used the tool. The three teams tended to create cards in the center of the workspace to start and then moved the cards to the edges of the display once they finished discussing the cards. Two of the teams resized the cards manually to a smaller size to make the most use of the space available while the same team that used the click and drop approach to create the cards used the collapse feature. Following the creation of five or six cards the teams started to create cards in the available space rather then increase the size of the iteration or backlog objects beyond the visible screen space.

During the discussions regarding work completed in the previous iteration, teams organized cards differently. UK-One used two iterations to denote which cards were complete and which ones were not. The UK-Two used the "Complete Card" feature to denote completed cards prior to creating a new iteration for the next iteration and moving cards from one to the other. The Victoria team made use of the backlog for completed stories and an iteration for incomplete and new ones. The Victoria team made use of the delete feature after breaking up a newly-created story card into two smaller cards. When breaking up cards, the other teams created a discard pile or edited the existing cards.

All three teams experienced trouble when editing cards in DAP. Team members were using a double click action to edit a field rather than the single click required. Teams experienced various amounts of time to figure this out with one of the teams having to ask the researcher how to accomplish the task. Once discovered, teams were

able to adapt quickly to the approach. Another problem was that some of the teams had trouble with creating cards. This happened when teams attempted on more than one occasion to create a card outside the product backlog or iteration objects[2].

Once teams were able to overcome the challenge of editing a story card, they were excited by the feedback provided by the live text feature. This was particularly noticeable with the Victoria group when the customer took control of the planning meeting and the creation of cards. A developer commented with enthusiasm "That's kinda neat, we can see you typing!"- P24. Other teams expressed similar sentiments.

Telepointers were used by all the teams for pointing to cards during the negotiation phase of the meetings or when team members were unsure of the meaning of some text. UK-One made significant use of the telepointers to point out the cards that they were referring to during their conversations and when they had made a change to a given story.

Communication throughout all three planning meetings using DAP was more focused than the previous planning meetings. Team members observed were less likely to have side conversations or become distracted by something else. During the first set of meetings, this was seen by developers passing notes amongst themselves and using facial expressions to express their personal opinions to the others. This was not the case during the use of DAP as the teams were focused on the display and the interactions by both sides.

---

[2] Continued development efforts on DAP now supports story card creation on the background workspace.

6.5.3 Feedback

Feedback from interviews, of those that participated in the study, was encouraging. Of the twenty-seven participants, twenty-one participated in the interview portion of the study. All those who were interviewed indicated that they liked the tool, thought that it was easy to use and thought that it was a significant improvement over telephone only distributed planning.

6.5.3.1 Usability

Participants found DAP easy and quick to use. Commenting "The software was quite user friendly and easy to operate and understand."- P20 and that they thought "… that its very, very easy to use."- P26. Teams found the learning curve to be almost non-existent. In all of the planning meetings where DAP was used, teams took seconds to become familiar with the tool's environment and start planning. Comments from participants indicating how they liked how quick they could start using, for example: "the learning curve was really quick… the application was laid out very nicely for that." – P24 the tool and how having the tool following the card metaphor made getting started with the tool easy.

6.5.3.2 Tool Interaction and Use

Participants' comments regarding interacting with the tool were very encouraging. The respondents found that the visual representations of story cards and iterations helped them in visualizing the project, commenting "It gave us an exact picture of the planning project" – P20. Some found the ability to simultaneously create cards and drag them at

will, similar to what they would find in a face-to-face environment was very important. During the planning meetings, simultaneous card creation and interaction rarely occurred, with only the UK-Two Team working simultaneously to create and edit cards. Many of those interviewed indicated that they liked the feedback from the telepointers and the live text. Commenting that they liked how "[they] could monitor whatever was happening on the screen of the other party"- P20 in near real-time and that "simple pointing [was] possible"- P15. Others found that just having some structure for the story cards was important in that it encouraged them to add more information to the card rather than just the story name. When using paper cards a participant commented that "… there was nothing that forced me to enter a description but the tool had the description so I felt like, you know, I should fill this in" – P23.

Feedback from the participants included some shortcomings found in DAP. One obstacle that everyone pointed out was related to editing the card. They were expecting that a double click would trigger an event that would allow them to edit the card. When asked about what made them want to try a double click to edit the card, many of the responses were "… in Windows, which is what most people use, you have to double click on something to change the name."- P17 or "I just assumed that it would be but I must have been confused with something else that I had worked on" – P23. A second issue that was raised was related to the prioritization of stories. Many participants felt that allowing for equal priority would be beneficial as some of the tasks were, in their minds, equally important. Others commented that they wished they had more of a visual indication of priority:

> "I was bummed that I didn't remember that. I knew that like higher up you put the cards the more priority they have and I completely forgot about that, maybe I needed something in the tool to remind me of that cause I think we could have used that a bit."- P23

The last major point that was highlighted by almost everyone was the fact that there wasn't enough screen real estate to plan the way they wanted. "There simply wasn't the room to put [the cards] together and cluster them the way we wanted" –P25. Most commented that they would have preferred a way to see more of the planning space at the same time. Most indicated that some sort of zoom functionality would have helped solve the problem:

> " I was trying to move all the cards and I felt like I wish that [the planning space] was bigger… I wish that you could zoom in and out. The problem is that when you zoom out you cant read the cards anymore" – P17

Others commented that it was hard to brainstorm to collect requirements because the participants were too restricted by the imposed hierarchy. They commented that they would have liked to be able to create cards directly on the workspace and not have to bother with iteration or the product backlog until they were ready to schedule and actually plan the project[3].

6.5.3.3 Communication

Communication was a dominant factor in the feedback from all the interviewees. Those interviewed felt that using the tool helped with guiding the communication that took place during the planning. They felt that during the distributed planning sessions

---

[3] Creating Story Cards on the workspace is now supported.

where no tool was used they, and their team members, tended to go off topic or run with an idea. As we saw in the previous section, where one of the customers was fixated on a topic and explaining the feature, there was no way for the developer to interject and ask a question. That developer indicated that they were very frustrated by that situation. The participant commented: "… you would normally be able to give them some sort of visual and just be like, ok wait!"- P11.

Body language was another topic where participants found that the distributed planning as a whole was challenging. Teams found that without body language cues, communication was challenging because "…you might consider yourself communicating properly with the client and getting the kind of answers you want when indeed you're not."-P24. This lack of body language and visual cues during the first round of planning in the use study teams led to some of the teams feeling like there was some segregation between the teams. In fact these teams did not feel like one team but two:

> "…we had our side and then there was their side and so we were talking amongst ourselves or exchanging notes on our side…I felt a strong alliance with our side and I felt like a separation from there side" – P23

The segregation between the two teams resulted in two issues. First, the teams found that they had a difficult time becoming comfortable interacting with someone they had never met.

> "Talking to a telephone with people on the other end I could not see was difficult. Normally you shake hands, have a coffee together or smile at each other. Do some small talk… it was kind of an awkward feeling in the beginning" - P12

The second issue was that teams were using an extremely high amount of facial expressions to express themselves to the others at their location. This was seen as both positive and negative. Some participants felt that this was somewhat unprofessional as they felt this wouldn't occur in a face-to-face situation. Other participants felt that the barrier created by the telephone prevented developer's expressions from influencing the customer. For instance:

> "… when they start making faces that subconsciously they're thinking that's the stupidest thing I've ever heard, at least those facial expressions are not being broadcast to the client who then might think that it is the stupidest thing I've ever heard when it could just be [a developer] not wanting to step out of line with what he knows…" - P19

The introduction of DAP didn't change people's desire to have body language to augment the communication. However, the respondents did find that using the tool brought the teams closer and alleviated the feelings of two separate teams.

> "… that one guy in particular, I really felt like he started out sort of watching and then got into using it and wanted to use it and as the meeting progressed. I thought it was a good thing cause it kind of kept him involved and because we're working on the same tool,I felt less of a separation." - P23

Everyone mentioned a video link to resolve the issue of sharing body language with the distributed colleagues. However, after mentioning it about half of those interviewed indicated that they would not feel comfortable with a video link and conceded that the loss of body language might be a necessary trade off to maintain the level of comfort that they had experienced.

The perceived productivity of the meetings was considered to be increased following the introduction of DAP. Participants found that, with the improved

communication in addition to being able to see and read the contents of the story cards, what they considered to be productivity was improved. Participants felt more confident that they better understood what they had to accomplish in the iteration "… people could see what was there, they could move the cards around [and] they got more of a feeling of the iteration" – P24. The amount of time spent discussing the stories was reduced, and creating cards was just as quick and easy as creating paper cards. The only point that participants acknowledged was increased in the first meetings, was the number of story cards created. However, they all indicated that they didn't consider the number of story cards to be an indication of productivity.

6.5.3.4 Planning with DAP after the study

Of particular interest to this research was that following the DAP case study the team continued using DAP for their project planning. During an interview with one of the participants in this case study it was found out that some technical issues occurred causing the DAP server and client to fail (the issue has since been resolved). The failure resulted in the case study team to switch mid-planning from using the tool back to using telephone and paper cards. The team found this very frustrating with one participant commenting:

> "…you noticed right away that it felt like one of your legs was missing and you're just like fumbling around with cards on the table speaking into the phone again … I noticed that it was a lot more awkward this time after using the tool successfully a couple of times." - P4

The team was able to complete the planning meeting; however, the perceived productivity was affected and the team's ability to plan was hindered.

6.6 Independent Feedback

During the last phase of this research, a researcher from the National Research Counsel of Canada (NRC) contacted the researcher about potentially using DAP as part of the infrastructure to support a distributed collaboration session. They were looking for a tool "…to emulate a remote and shared wall-size poster on which you can put and move post-it notes."-Researcher. The purpose of the post-it notes was not limited to story cards and would be used for business goals or user personas. They were evaluating DAP against CardMeeting [CardMeeting 2007] to determine which tool better fit their needs.

After a two-week evaluation, they concluded that DAP was better suited for release and iteration planning; however, regarding the requirements gathering sessions, DAP was not ideal.

The group ultimately decided to use CardMeeting for their distributed collaboration and provided a number of reasons for their choice. The main reason for not using DAP was the limited support for brainstorming and requirements gathering. They felt that DAP was too restrictive by requiring a backlog or iteration to create story cards. They needed a tool that let team members create cards anywhere in the shared workspace. Also, as their tasks were not limited to story creation, they felt that the story card objects contained too much additional information that was not necessary for their needs.

As they would be using the tool for requirements gathering in addition to defining business goals and user personas, they needed to be able to distinguish quickly between the different types of artefacts on the screen. As DAP only supports one type of card and

does not provide colour to distinguish the cards this requirement could not be met by DAP. The last requirement necessary for their needs was a way to see a complete overview of the workspace at once[4]. As the ability to zoom in or out is not supported in DAP, this was yet another reason to choose another alternative.

The overall feedback from the researcher at the NRC, although they didn't choose to use DAP, was positive. Although they chose to use a different tool as it better fit their needs, they did indicate that they would be "…try[ing] AgilePlanner for [their] iteration and release planning meetings…"-Researcher.

6.7 Limitations

The observations conducted and the feedback gathered from the participants provides insight into the effects of Distributed AgilePlanner. The results presented here, though promising, contain the following threats to its validity.

First, all participants in the user study and most of the participants in the case study were graduate students at various academic institutions in Canada and the United Kingdom, only one participant was an industrial developer. Participants generally had minimal practical exposure to agile methods and agile planning and as such cannot be truly representative of the target population.

Second, the participation in the research was voluntary. Teams involved in the user studies were volunteers; those participating the observation portion of the case studies were voluntary at the group level, feedback was voluntary on an individual basis.

---

[4] Continued development efforts on DAP now provide a zoom feature.

In essence, those that completed both the observation and feedback phases of the research were self-selecting and that introduces a user bias to the results.

Third, many of the participants that provided feedback were part of the same research group and laboratory as the observer and interviewer. This could have resulted in a bias being introduced into the feedback provided by these participants.

Fourth, observations and interviews were conducted and analyzed by the same researcher. This researcher is the primary developer for DAP and will inevitably introduce a research bias onto the interpretation of the observations and feedback.

Fifth, the number of teams participating in the research is quite small and as such provides limited insight onto which generalizations can be based. Therefore, generalized conclusions are questionable at best.

Sixth, industrial evaluation is minimal as only one case study was near to industry. Developers involved in industrial and academic settings have different outlooks and goals. Findings from this research could represent industrial developers improperly.

Seventh, evaluation focused on comparing one existing method of distributed planning against DAP. Limiting the comparison point for the participant can positively bias the results, as some may not have other points of reference. Limiting the research to only comparing the two approaches was due to time restrictions (two years) imposed on this research.

6.8 Summary

This chapter presented the goals, methodology and results of two case studies, three user studies and one independent evaluation of DAP. The results of this preliminary

evaluation show that DAP provides an improvement over phone and paper index card planning by providing not only shared access to card information but additional awareness information that helps alleviate some of the communication concerns that can occur. Although preliminary in nature, the findings from this evaluation are encouraging and suggest further development and investigation is warranted.

Chapter Seven: Conclusion

At the beginning of this research, in Chapter One, the challenges experienced by existing distributed agile teams were presented. It was proposed that, in order to better support distributed agile teams, a tool that supports card based planning was necessary. We presented development of a card-based distributed planning tool that supported creation and organization of planning artefacts with near real-time updating of the planning data in a way that was intuitive and natural. To determine the effectiveness of the proposed tool a preliminary evaluation was conducted.

7.1 Thesis Contributions

The goals in this work are to determine how to better support card-based planning for distributed agile teams and to determine the effectiveness of a distributed card-based planning tool. To implement the goals, the following three contributions have been made.

Tool Requirements. The first contribution made by this research was the derivation of sixteen requirements for creating a distributed, card-based planning tool. The requirements, presented in Chapter Two, combined features and requirements from existing agile planning tools and research from the research area of agile methods. In addition, research from the groupware community provided additional insight into requirements necessary for effective tool support for group collaboration.

Distributed AgilePlanner. Following the requirements extracted from existing work, Distributed AgilePlanner was developed to support distributed agile teams. Although, DAP doesn't fully support all of the requirements identified, it is an

improvement over existing tool support. DAP is a strong attempt to support creating and organizing planning artefacts in a shared workspace where planning information is updated in near real-time. DAP allows for more intuitive and natural interactions between the team members and between the individuals and the artefacts.

Evaluation. The third contribution of this research is a preliminary evaluation of Distribute AgilePlanner. The evaluation was conducted in three different ways: First, case studies of two development teams using DAP during part of their project planning meetings. Second, three user studies where individual volunteers, from Canada and the United Kingdom, were organized into teams and simulated distributed card-based planning. Third, an independent evaluation of DAP against another tool, where the primary usage was not project planning but distributed brainstorming activities. Results from all three evaluations were encouraging; with a strong majority of the participants indicating that using DAP was an improvement over using paper-based planning for distributed teams.

7.2 Future Work

Distributed AgilePlanner has shown promise in its ability to improve support for distributed card-based planning. However, there is always room for improving the tool and augmenting research.

Since the evaluations described earlier, development efforts on DAP have continued following the initial tool construction. Developers collaborating with a software firm have since taken over construction and maintenance of DAP to allow for its

release to the general public. Their efforts have contributed to improving some of its shortcomings highlighted during the evaluation.

7.2.1 New Features

The bulk of the new features introduced to DAP have been to improve the link between how individuals conduct index card-based planning and how DAP simulates card-based planning.

The first major change to DAP was the elimination of the forced hierarchy. This allows teams to use the shared workspace for brainstorming without the need for creating a product backlog. This added freedom allows for DAP to cater to various interpretations of iterative planning in addition to supporting brainstorming activities.

The second addition to DAP is the support of different card colours for indicating differences in card type. Participants indicated during the evaluation that this would be a beneficial addition to the tool. The addition of distinguishing card colours allows for DAP to fulfill yet another requirement identified in Chapter Two.

Throughout the evaluation of DAP, screen real estate often came up as a limiting factor to its adoption. To alleviate some of the strain that a limited screen size presents, DAP has been augmented with support for zooming in the workspace. This allows for teams to change the amount of the workspace visible to better support their needs.

7.2.2 Future Developments

Continued work on DAP has allowed the planning tool to fulfill additional requirements then those originally presented. However, there are still some shortcomings.

There are still a few of the requirements that are as yet unfulfilled. First, there is no authentication mechanism preventing unauthorized access to the planning data. This requirement is necessary for wide-scale adoption where the project being developed is not common knowledge to the general public. Second, improving change notification to include notifying offline team members could increase the amount of information transfer to team members. Increasing information transfer could allow team members not present to be up to date with any changes to the project. Third, awareness information is still limited for work occurring outside the visible workspace. Providing radars to augment the zooming features could greatly increase the collaboration amongst subgroups. Fourth, DAP remains a single-user application, allowing for only one individual to interact with any given client. Distributed development teams, while collaborating, are highly likely to have more then one individual located at each site. However, with DAP in its current state teams are required to take turns interacting with the workspace or to appoint an individual to act as a scribe for the meeting. Augmenting DAP to allow for simultaneous interactions at a given site would alleviate this turn-taking and allow for interactions that more closely simulate collocated planning where anyone at anytime can interact with the planning artefacts.

7.2.3 Further Evaluation

Further evaluation into how DAP affects distributed planning is needed. As Chapter Five showed, there were a number of limitations present in the initial evaluation. Specifically, there is a need to extend the evaluation to include a larger number of industrial participants to verify the results presented here. In addition, further evaluation

needs to compare DAP against other planning tool alternatives. All future evaluations need to be longitudinal in nature in order to examine the long term effects of a tool of DAP's nature.

References

[AgileManifesto 2001] AgileManifesto. (2001). "Manifesto for Agile Software Development." Retrieved Sept 15, 2007, from http://agilemanifesto.org/.

[Beck 2000] Beck, K. (2000). Extreme Programming Explained: Embracing Change, Addison Wesley

[CardMeeting 2007] CardMeeting. (2007). "CardMeeting." Retrieved Sept 24, 2007, from http://www.cardmeeting.com/.

[Cohn 2006] Cohn, M (2006). Agile Estimating and Planning, Pearson Education Inc

[Danube 2007] DanubeTechnologies (2007). "ScrumWorks Basic " Retrieved Sept 24, 2007, from http://www.danube.com/scrumworks/basic.

[Dyck et.al;. 2004] Dyck, J., C. Gutwin, S. Subramanian and C. Fedark (2004). High-Performance telepointers. ACM conference on Computer-Supported Cooperative Work (CSCW'04).

[Ellis et.al. 1991] Ellis, C. A., S. J. Gibbs and G. Rein (1991). "Groupware: some issues and experiences." Communications of the ACM 34(1-Jan 1991): 19.

[EBE 2007] E-Business Engineering (2007). "Mase." Retrieved Sept 24, 2007, from http://ebe.cpsc.ucalgary.ca/AgilePlannerWebUI/index.html.

[Eclipse Foundation 2007: Eclipse] Eclipse Foundation (2007). "Eclipse." Retrieved Sept 29, 2007, from http://www.eclipse.org/.

[Eclipse Foundation 2007: Graphical Editing Framework] Eclipse Foundation (2007). "Eclipse Graphical Editing Framework." Retrieved Sept 29, 2007, from http://www.eclipse.org/gef/.

[Eclipse Foundation 2007: Rich Client Platform] Eclipse Foundation (2007). "Eclipse Rich Client Platform " Retrieved Sept 29, 2007, from http://wiki.eclipse.org/index.php/Rich_Client_Platform.

[Floranta 2007] Floranta (2007). GlueWiki: Wiki based card tool

[Greenberg 1990] Greenberg, S. (1990). "Sharing views and interactions with single-user applications." ACM SIGOIS Bulletin 11(2-3): 10.

[Grudin 1994] Grudin, J. (1994). "Groupware and social dynamics: eight challenges for developers." Communications of the ACM 37(1): 13.

[Gutwin et.al. 1996] Gutwin, C. and S. Greenberg (1996). Workspace awareness for groupware. Conference companion on Human factors in computing systems (CHI'96), Vancouver, British Columbia Canada, ACM press.

[Gutwin et.al. 1998] Gutwin, C. and S. Greenberg (1998). Effects of awareness support on groupware usability Conference on Human Factors in Computer Systems (SIGCHI), Los Angeles, California, United States, ACM Press/ Addison-Wesley Publishing Co.

[IBM 2005] IBM (2005). Eclipse Draw2D API.

[Ishii et.al. 1993] Ishii, H., M. Kobayashi and J. Grudin (1993). "Integration of interpersonal space and shared workspace: ClearBoard design and experiments." ACM Transactions on Information Systems (TOIS) 11(4): 26.

[Kelter et.al. 2003] Kelter, U., M. Monecke and M. Schild (2003). "Do we need agile software development tools?" Lecture notes in computer science 2591: 18.

[Lauwers et.al. 1990] Lauwers, J., T. Joseph, K. Lantz and A.Romanow (1990). Replicated architectures for shared window systems: a critique. ACM SIGOIS and IEEE CS TC-OA conference on Office information systems, Cambridge, Massachusetts, United States, ACM Press.

[Liu 2005] Liu, L. (2005). An Environment for Collaborative Agile Planning Computer Science Calgary University of Calgary Masters of Science: 118.

[Malone et.al. 1992] Malone, T.W., Lai, K.Y. Toward intelligent tools for information sharing and collaboration. In R. P. Bostrom, R. T. Watson & S. T. Kinney(Eds). Computer Augmented Teamwork: A guided Tour, New York: Van Nostrand Reinhold, 1992.

[Mandviwalla et.al. 1994] Mandviwalla, M. and L. Olfman (1994). "What do groups need? A proposed set of generic groupware requirements." ACM Transactions on Computer-Human Interaction (TOCHI) 1(3): 23.

[Maurer 2002] Maurer, F. (2002). "Supporting Distributed Extreme Programming." Lecture notes in computer science 2418: 19.

[Microsoft 2007:Excel] Microsoft. (2007). "Microsoft Excel." Retrieved Sept 24, 2007, from http://office.microsoft.com/en-us/excel/FX100487621033.aspx.

[Microsoft 2007:NetMeeting] Microsoft. (2007). “Microsoft NetMeeting” Retrieved Oct 24, 2007, from http://www.microsoft.com/downloads/details.aspx?FamilyID=26c9da7c-f778-4422-a6f4-efb8abba021e&DisplayLang=en

[Rally 2007] Rally Software. (2007). "Rally's Agile Life Cycle Management Solutions." Retrieved Sept 24, 2007, from http://www.rallydev.com/products.jsp.

[Rees 2002] Rees, M. J. (2002). A feasible user story tool for agile software development? Ninth Asia-Pacific software engineering conference (APSEC'02), IEEE: 22.

[Reeses et.al.2004] Reeses, M. and J. Zhu (2004). "Moomba - A collaborative environment for supporting distributed extreme programming in global software development." Lecture notes in computer science 33092: 12.

[Reinhard et.al.1994] Reinhard, W., J. Schweitzer, G. Volksen and M. Weber (1994). "CSCW Tools: Concepts and Architectures." Computer 27(5): 8.

[Schummer et.al.2001] Schummer, T. and J. Schummer (2001). Support for distributed teams in extreme programming. Extreme programming examined. Boston, MA, Addison-Wesley Longman Publishing Co., Inc.

[Skype 2007] Skype. (2007). "Skype Features." Retrieved Sept 25, 2007, from http://skype.com/intl/en/features/.

[Smart Technologies 2007] Smart Technologies (2007). "SMART Board Interactive Whiteboards." Retrieved Sept 23, 2007, 2007, from http://www2.smarttech.com/st/en-US/Products/SMART+Boards/default.htm.

[Stefik et.al. 1987] Stefik, M., D. G. Bobrow, G. Foster, S. Lanning and D. Tatar (1987). "WYSIWIS revised: Early experiences with multiuser interfaces." ACM Transactions on Information Systems (TOIS) 5(2): 20.

[Sun 2007] Sun Microsystems (2007). "Java " Retrieved Sept 24, 2007, from http://java.sun.com/.

[Tang 1991] Tang, J. C. (1991). "Findings from observational studies of collaborative work " International Journal of Man-Machine Studies 34(2): 17.

[Thoughtworks 2007] Thoughtworks (2007) “Mingle” Retrieved Oct 25, 2007, from http://studios.thoughtworks.com/mingle-project-intelligence

[VersionOne 2007] VersionOne. (2007). "Agile Project Management Tools." Retrieved Sept 24, 2007, from http://www.versionone.com/products.asp.

[Wiki.org 2002] Wiki.org. (2002, June 27, 2002). "What is Wiki." Retrieved Aug 6, 2007, from http://www.wiki.org/wiki.cgi?WhatIsWiki.

[Wikipedia 2007: Wikipedia] Wikipedia. (2007). "About Wikipedia." Retrieved August 6, 2007, from http://en.wikipedia.org/wiki/Wikipedia:About.

[Wikipedia 2007: NetMeeting] Wikipedia. (2007). "Microsoft Netmeeting." Retrieved Sept 24,2007, from http://en.wikipedia.org/wiki/Microsoft_NetMeeting.

[XPlanner 2007] XPlanner. (2007). "XPlanner Overview." Retrieved Sept 24, 2007, from http://www.xplanner.org/.

APPENDIX A: ETHICS APPROVAL

UNIVERSITY OF CALGARY

MEMO

CONJOINT FACULTIES RESEARCH ETHICS BOARD
c/o Research Services
Main Floor, Energy Resources Research Building
3512 - 33 Street N.W., Calgary, Alberta T2L 1Y7
Telephone: (403) 220-3782
Fax: (403) 289 0693
Email: bonnie.scherrer@ucalgary.ca
Thursday, December 14, 2006

To: **Robert E. Morgan**
Computer Science

From: Dr. Janice P. Dickin, Chair
Conjoint Faculties Research Ethics Board (CFREB)

Re: **Certification of Institutional Ethics Review:** Examination of the Impact of Software Tools to Support Distributed Agile Project Planning

The above named research protocol has been granted ethical approval by the Conjoint Faculties Research Ethics Board for the University of Calgary.

Enclosed are the original, and one copy, of a signed **Certification of Institutional Ethics Review**. Please make note of the conditions stated on the Certification. A copy has been sent to your supervisor as well as to the Chair of your Department/Faculty Research Ethics Committee. In the event the research is funded, you should notify the sponsor of the research and provide them with a copy for their records. The Conjoint Faculties Research Ethics Board will retain a copy of the clearance on your file.

Please note, an annual/progress/final report must be filed with the CFREB twelve months from the date on your ethics clearance. A form for this purpose has been created, and may be found on the "Ethics" website, http://www.ucalgary.ca/UofC/research/html/ethics/reports.html

In closing let me take this opportunity to wish you the best of luck in your research endeavor.

Sincerely,

Russell Burrows
For:
Janice Dickin, Ph.D., LLB., Faculty of Communication and Culture and
Chair, Conjoint Faculties Research Ethics Board

Enclosures(2)
cc: Chair, Department/Faculty Research Ethics Committee
Supervisor: Frank Maurer

UNIVERSITY OF CALGARY

CERTIFICATION OF INSTITUTIONAL ETHICS REVIEW

This is to certify that the Conjoint Faculties Research Ethics Board at the University of Calgary has examined the following research proposal and found the proposed research involving human subjects to be in accordance with University of Calgary Guidelines and the Tri-Council Policy Statement on *"Ethical Conduct in Research Using Human Subjects"*. This form and accompanying letter constitute the Certification of Institutional Ethics Review.

File no:	**5048**
Applicant(s):	**Robert E. Morgan** Frank Maurer
Department:	**Computer Science**
Project Title:	**Examination of the Impact of Software Tools to Support Distributed Agile Project Planning**
Sponsor (if applicable):	

Restrictions:

This Certification is subject to the following conditions:

1. Approval is granted only for the project and purposes described in the application.
2. Any modifications to the authorized protocol must be submitted to the Chair, Conjoint Faculties Research Ethics Board for approval.
3. A progress report must be submitted 12 months from the date of this Certification, and should provide the expected completion date for the project.
4. Written notification must be sent to the Board when the project is complete or terminated.

Janice Dickin — 13 December 2006

Janice Dickin, Ph.D, LLB, **Date**
Chair
Conjoint Faculties Research Ethics Board

Distribution: (1) Applicant, (2) Supervisor (if applicable), (3) Chair, Department/Faculty Research Ethics Committee, (4) Sponsor, (5) Conjoint Faculties Research Ethics Board (6) Research Services.

APPENDIX B: USER STUDY PROBLEM DESCRIPTION

Project Overview.

Grapefruit Computers has been hired to develop a point of sale software for pubs and bars. Your team's job is to guide your development team on how to produce a quality product that meets your end user's needs.

Your team's job is to come up with specific requirements for your developers. Your research department in conjunction with local bar and pub owners have provided the following (vague) guidelines.

The system must:

- Be easy and quick to use,
- Work in low light environments,
- Allow for easy changes to inventory and pricing.

Additional Information:

You will be following an agile approach to software development. Your requirements will need to be in the form of user stories. Your development team will have a total of 40 hours per person per week. You need to make sure that they do not estimate more then they can handle.

APPENDIX C: INTERVIEW OUTLINE

Question 1: What advantages did having teams in multiple locations have on the planning of your project?

Question 2: What disadvantages did having teams in multiple locations have on the planning of your project?

Question 3: What was the hardest issue to overcome when some of the members of your team were not physically present at the meeting?

Question 3a: How did you overcome this issue?

Question 4: How would you improve the method of planning to better accommodate team members in multiple locations?

Question 5: What would you keep the same with respect to the way you conducted the planning meeting?

Question 6: Do you feel that having distributed team members affected the productivity of the planning meeting?

Question 7: Do you feel that you were able to participate in the planning meeting to the extent that you wanted?

Question 7a: If not, what would have allowed you to participate to the extent that you wanted?

Question 8: Did having team members in multiple locations affect your feeling of belonging to the team? How?

The following questions are specific to the Distributed AgilePlanner tool.

Question 9: What features do you like most about Distributed AgilePlanner? Why?

Question 10: What features do you like least about Distributed AgilePlanner? Why?

Question 11: Did Distributed AgilePlanner address the issues you found hard to overcome during the last planning session? How?

Question 12: Did using Distributed AgilePlanner make you feel more or less inclined to participate in the planning meeting than you did during the last planning meeting? Why?

Question 13: How would you improve Distributed AgilePlanner to better suit your personal needs?

Question 14: How would you improve Distributed AgilePlanner to better suit the needs of your project?