

Tools for Supporting Distributed Agile Project Planning

Xin Wang, Frank Maurer, Robert Morgan and Josyleuda Oliveira

Abstract Agile project planning plays an important part in agile software development. In distributed settings, project planning is severely impacted by the lack of face-to-face communication and the inability to share paper index cards amongst all meeting participants. To address these issues, several distributed agile planning tools were developed. The tools vary in features, functions and running platforms. In this chapter, we first summarize the requirements for distributed agile planning. Then we give an overview on existing agile planning tools. We also evaluate existing tools based on tool requirements. Finally, we present some practical advices for both designers and users of distributed agile planning tools.

1 Introduction

Agile project planning is an important activity for agile teams. It allows a team to start focusing on the next development iteration and drives the evolution of software products. The goals of agile project planning include:

- Controlling the software development progress.
- Kicking off a new development iteration and planning tasks for it.
- Providing a focal point for the communications between developers and customers
- Enhancing the collaborations within software development teams.

Xin Wang
Ivrnet Inc., Calgary, Alberta, Canada. e-mail: x.wang@ivrnet.com

Frank Maurer and Josyleuda Oliveira
Department of Computer Science, University of Calgary, Calgary, Alberta, Canada. e-mail: {maurer,oliveirj}@cpsc.ucalgary.ca

Robert Morgan
Red Duck Solutions, Calgary, Alberta, Canada. e-mail: robert.morgan@redducksolutions.com

Project planning benefits agile development in both management and communication aspects: it reviews project progress, detects development bottlenecks and generates sound plans to decide on the use of team resources. It connects developers with customers and reduces the misunderstandings between each other.

Traditionally, agile planning is conducted in a co-located environment. Participants are situated at the same site, using face-to-face communication to plan future tasks. Index cards are the major artefact that supports the project planning process. New tasks are created by writing new cards. Tasks are prioritized by sorting the cards.

When we observed co-located agile project planning meetings, we found three primary factors that positively affect their quality:

- *shared access index cards describing tasks,*
- *flexible use of index cards and*
- *easy interactions among meeting participants.*

The shared access to index cards enables participants to understand the current state of the planning process. The flexibility of using index cards helps developers plan the project in a convenient manner. Easy interactions among participants improve collaboration among all stakeholders. The collaborative environment improves the effectiveness of project planning and help shape the group into a unified and well-communicating team.

In a co-located environment, all three primary factors are easily provided. Paper index cards on a table provide an intuitive and shared access to project tasks. Physical cards are easily edited or ranked. Natural interactions are the result of verbal communications and body gestures.

However, when agile teams are distributed it is difficult to conduct traditional agile project planning meetings. Sharing planning artefacts among spatially-separated environments becomes challenging, and interaction among planning participants are more difficult. When we conducted interviews with distributed agile teams from Brazil and Canada, several issues were pointed out. These include:

- Making decisions becomes much harder than that in co-located project planning meetings.
- Sites were not talking like a unified team.
- There is less communications within a distributed team than a co-located one. Respectively, problems are not reported until they are bigger. Then, they require more time and money to solve them.
- Misunderstandings are raised and the chance of rework is increased significantly.

A first commonly used approach for distributed project planning is to utilize audio and/or video conferences with paper index cards at each site. Telephones and cameras are employed to set up synchronous verbal and visual communication among different sites. Although such an approach establishes remote interactions, index cards are not shared in this context they either reside at one site only or are replicated manually to the other sites. Story cards from one site are not directly shown to the distributed teams, and key behaviours, such as modifying index cards

are difficult to share with remote colleagues. When distributed teams replicate paper story cards to each site, the duplication increases the risk of misunderstandings. To understand the impact of distributed teams, we conducted a small scale experiment. We observed a meeting between teams in Canada and United Kingdom that used conference calls and replicated paper index cards for the meeting. The discussions were often interrupted by both teams realizing they were not talking about the same card. The meeting also ended with both sites generating a different number of index cards for the same topic, which represents a severe misunderstanding between both sites. Anecdotal evidence from practitioners confirm these findings.

As a result of these experiences, several attempts were made at using computers and the Internet to set up a card-centered project planning environment. A large number of tools are now available to support distributed agile planning. In this chapter, we start by discussing high-level requirements for distributed agile planning tools. We then review existing tools based on these requirements. Practical advice will be provided for users and designers of distributed planning tools to assist with selecting or designing an appropriate tool to support distributed agile planning.

2 Distributed Planning Tool Requirements

Our study started by setting up a series of requirements for distributed agile planning tools. The requirements are used as criteria to evaluate and compare existing tools. From these requirements, users can evaluate the benefits and limitations of tool usage within their teams. Designers will also understand from which aspects a distributed agile planning tool can be improved. In this section, we break down the tool requirements to those specific to agile planning and those specific to collaborative interactions:

- *Agile planning requirements:* Distributed agile planning tools are specifically designed to support agile developments. Therefore, the primary requirements are to cover the major functions prescribed by the agile planning processes such as creating and editing index cards. The agile planning requirements concentrated on the tools functional capabilities and determine whether and how much agile planning is supported. They are proposed from reviewing literature from previous research [1] [3] and our practical experiences on developing agile planning tools [9] [12] [15].
- *Requirements for collaborative interactions:* Agile project planning is essentially a group-based collaborative activity. It can be improved by providing easy interactions for team members. Unhindered interactions are also the main approach that makes distributed collaborations more effective. In order to improve the interactions among distributed groups, a substantial amount of research was conducted on computer supported collaborative work (CSCW) processes and groupware. We believe the general studies on CSCW and groupware largely benefit the development of distributed agile planning tools. In this section, we referred to the literature on groupware research and proposed a series of requirements for

collaborative interactions. The requirements show a higher level goal than just being able to do agile project planning. These requirements concentrate on tool usability and highlight the importance of supporting interpersonal interactions by enabling intuitive human-computer interaction for distributed agile planning. They also indicate how a distributed agile planning tool can be effective and convenient to use.

2.1 Agile Planning Requirements

The agile planning requirements stem from the need to create, organize and share planning information. Having observed distributed agile planning as well as analysing the related literature, we found the following requirements are critical to support distributed agile planning. The first three requirements are derived from previous studies made by Abrahamsson et al. [1] and Larman [3]. The remaining requirements are a result of our observations of planning meetings and experiences in developing distributed agile planning tools.

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 artefacts. In a co-located scenario, a developer grabs an empty card and edits it to define a new task. He/she also can remove obsolete cards from the planning table. Remote agile teams still follow a card-centred planning process: given only an audio link, they create and manipulate index cards on their own site. Thus, operations for creating/editing/deleting cards are needed in any tool supporting distributed planning.
2. **Handle effort estimates.** 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 estimate new tasks. Managing this information can also be of use when determining the scope of current iterations (yesterday's weather).
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.
4. **Moving stories from one iteration to another.** Observing real-world agile teams has shown us practical cases where a story card is transferred to next iteration or moved into/out of the backlog. Respectively, distributed planning tools must provide a feature to support this behaviour.
5. **Authentication.** Security is important to prevent unauthorized access and modification to the information contained in the project plan.
6. **Real-time updates of the plan.** Remote access to the project artefacts is required such that, as changes occur, updated information is available on each participating site instantaneously.

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 is often used by teams.
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 and makes it easier to keep the plan up to date for progress tracking.

2.2 Requirements for Collaborative Interactions

The following requirements outline considerations for how a distributed agile planning tool can support interpersonal interactions and enhance collaboration within distributed team members. We determine requirements based on contributions by groupware researchers [2] [6] and combine them with our observations of distributed agile project planning meetings.

1. **Fluid transition between individual and collaborative work.** Systems need to support distinguishing private data from public data.
2. **Telepointers for pointing and gesturing.** Telepointers are a groupware technology that uses a remote mouse pointer to represent mouse movement happening on other computers. Telepointers allow remote team members to point to specific index cards, thereby increasing the shared understanding of the current discussion. They also allow teams to follow interactions happening on remote displays.
3. **Real-time information sharing.** Sharing information requires that changes to one workspace be updated in other workspaces instantaneously.
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.
5. **Joining and leaving meetings.** Team members should be able to connect and disconnect with ease and not affect others connected to the system.
6. **Fluid subgroup formation and dissolution.** For large scale projects consisting of teams of teams, subgroups need to be represented and supported by the planning tool. When supporting subgroup creation, the potential for isolation needs to be minimized in order for the subgroup to be informed of the other participants.
7. **Simultaneous interaction.** Supporting team members interacting simultaneously is important as it better simulates how teams interact in a co-located environment. Forcing teams to take turns would result in more overhead for the team during the planning meeting.

3 Tool Review

Software systems to support agile project planning in distributed environments have been available for some time. Some tools focus on documenting the outcomes of a planning meeting for progress tracking during the iteration. Others target to support the actual planning meeting. Unfortunately, this difference is often not highlighted in the relevant literature and marketing material.

To review existing agile planning tools, we first collected candidate tools that are published online, mentioned by our interview participants (industrial agile developers), introduced by our partner companies or described in the literature. We reviewed the tools and found that although existing tools showed some diversity, they could still be categorized by design goals, functionalities and supported platforms. The categorizations help to reveal common features, advantages and limitations of existing agile planning tools. Table 1 lists the basic categories and sample tools in our study. Some tools are found sharing feature of more than one category.

Table 1 Categories and sample tools.

Category	Sample Tools
Wiki	MASE, PMWiki, JSPWiki, MediaWiki
Web-form based application	Rally, VersionOne, ScrumWorks, XPPlanner, Mingle
Board-based application	CardMeeting, Gluewiki, AgilePlanner, MASE, Mingle, Danube
Plugins for IDE	IBM Jazz, Jira+GreenHopper, ProjectCards
Synchronous agile planning tool	DAP, CardMeeting
Tabletop-based agile planning tool	APDT

3.1 Wikis

Wiki-based agile planning tools utilize Web technologies to publish, manage, integrate and distribute agile planning information. The advantage of using Wiki-based systems is that they provide a plain environment, making it easy to check project status, update task lists and view the team members' work progress. Wikis are an *asynchronous* platform for agile developers' communication and, thus, mostly helpful for progress tracking. The following scenarios show how an agile development team can use them:

- Publishing story cards [11]: After a project planning meeting, new wiki pages will be created to publish all the card information. Software developers and project managers will be able to access the wiki pages and check their tasks.

- Story card management: software developers are responsible for accessing the wiki pages and updating their cards every day. Updating the card status facilitates managing the development progress.
- Sharing knowledge: a software developer can post his/her questions to a wiki, and his/her colleagues can view the questions to provide assistances. Meanwhile, one developer's experience of solving some critical problems can also be posted on the wiki to provide help to his/her teammates.

Wikis support asynchronous interactions for distributed teams. Using wiki pages does not rely on any specific software on the client side (any web browser will do). However, wikis only fulfill the minimum requirement of agile planning (creating/editing/deleting planning artifacts). Specific information of a user story, such as estimated hours, are mingled with plain text describing the story.

3.2 Web Form-based Applications

Designers of distributed agile planning tools realized the advantages (easy access) and limitations (loosely organized planning data) of wiki pages and started to use advanced Web technologies to create a series of Web form-based applications. Compared with plain text wikis, the structured data stored by such tools supports more sophisticated functions and more flexible operations to manipulate agile planning information.

Web form-based applications are often used for publishing and managing agile planning data. Such tools include commercial products like Rally, VersionOne, ScrumWorks, as well as open source products like XPPlanner. These applications use Web forms to create and manipulate planning data. They also set up basic workflows for sharing data amongst distributed agile developers. Figure 1 shows a screenshot of the Rally tool.

Figure 1 shows that agile planning data is more structured by Rally than in wiki. Using the tool, users can change the status of a story card by clicking the status button. They can also update the estimated work hours or descriptions of a task. Amongst other features, the Rally tool generates a burn down chart to help project managers and team developers understand progress of their projects.

The Rally tool shows some common features of Web form-based agile planning tools. Creating, editing and deleting story cards is supported. Charts are widely used to visualize the project progress. Agile planning data is well organized in projects, iterations, the backlog and story cards. Moreover, Web form-based applications can distinguish the roles of the users (such as developers or managers) and generate appropriate views for different user groups. The structured data managed in such tools provide semantically richer views on the agile process than text stored in wikis.

As Web technology is mature and the Web access is easily accepted by users, Web form-based applications dominate existing agile planning tools. However, most Web form-based tools are only for asynchronous usage (asynchronous data shar-

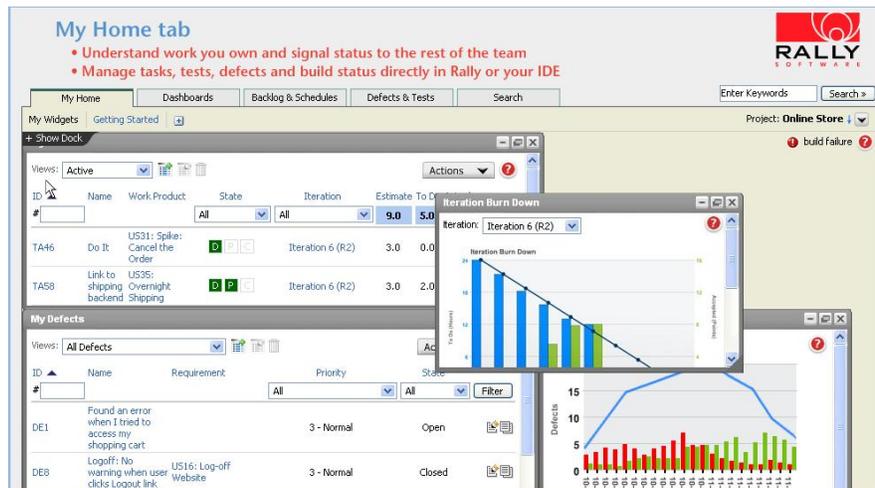


Fig. 1 Web form-based project planning tool[8]

ing, reporting, decision making, and daily card management), the synchronous agile planning, particularly the project planning meetings are not supported.

3.3 Card-Based Planning Systems

Card-based planning systems are systems that use visual representations that resemble index cards for representing tasks. These types of systems try to mimic physical card based planning. Glue Wiki, CardMeeting and AgilePlanner, amongst others, fall into this category. Several commercial agile planning tools, such as Thoughtworks Mingle, integrate a card-based planning with form-based tools. The benefit that card-based systems bring to planning is that they allow teams to interact with the cards as they are used from co-located settings.

Mingle (Figure 2) uses two-dimensional representations of index cards that teams can edit 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 and organize them spatially.

Card-based planning systems explore the collaborative interactions in distributed agile planning. The designs expect that showing the visual effects of “paper index card” might help agile teams adopting the tools. Card-based planning systems rely on spatial layout to make the current plan easier to understand. However, existing card-based planning tools (CardMeeting, Danube) do not show who is participating in the planning meeting. They specifically do not show if and who is currently interacting with planning artifacts. Knowing who is interacting with a planning artifact is important as it encourages communication and collaboration.



Fig. 2 Card-based system in Thoughtworks Mingle[13]

3.4 Plugin for Integrated Development Environment

Integrating project planning tools with Integrated Development Environment (IDE) will provide software developers a convenient environment for managing both codes and planning data, such as story cards. At present, we observed two types of plugins. The majorities are repeating the major functions of native/Web-based project planning tools, which allow for browsing and editing project planning data. While another type of plugin, besides showing and managing planning data, have tried to find and utilize the relation between user stories and the practical working codes. It enables users to connect high level story cards with low level test cases and codes. It bridges the logical gaps between the user requirements with developers' implementation. IBM Jazz [4] explores integrating project planning tools with software developing platform. It provides an Eclipse-based client to enable software developers mapping their story cards with specific source codes. Navigation between the codes and cards are also provided. Besides IBM Jazz, Microsoft Visual Studio Team System (VSTS) is also interested in introducing the project planning plugins to Visual Studio platforms. Other related project planning tools includes "Scurm for Team System", "Jira + GreenHopper", and "ProjectCards".

3.5 Synchronous Project Planning Tool

Although Web technologies allow sharing of agile planning data, the features of agile planning tools are still limited by the capabilities of Web browsers. Several requirements of collaborative interactions, such as synchronous notifications, and using telepointers for pointing and gestures, are not fulfilled. Moreover, by reviewing the practical needs of industrial agile developers, we found that synchronous agile planning meetings are weakly supported by such tools.

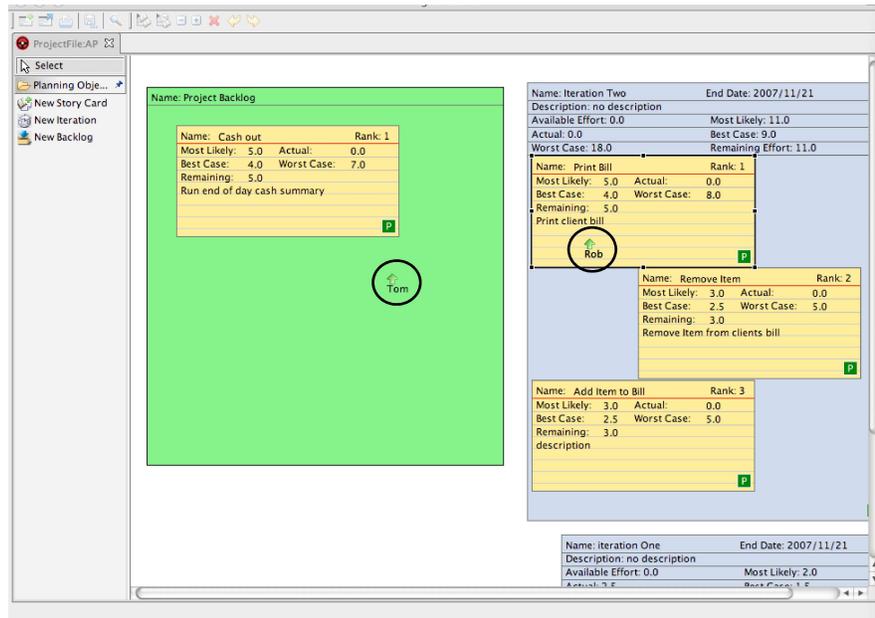


Fig. 3 Distributed Agile Planner with Telepointers and Digital Cards

Distributed Agile Planner (DAP) [9] is a standalone application for synchronous, card-based agile planning meetings. DAP mimics paper index cards. It simulates a whiteboard in a meeting room and utilizes electronic index cards to simulate paper index cards. The user interactions of DAP include creating, moving and deleting cards. To support distributed collaboration, DAP provides telepointers to represent mouse pointers of remote clients. The position of a tele-pointer is updated in real-time. Thus, a collaborator can understand his/her remote partner's mouse movement just like looking at hand movements in a traditional co-located meeting. Figure 3 shows the interface of DAP, on which a set of story cards and iterations are displayed. The green arrow is the "telepointer" which acts as a remote mouse pointer to indicate the focus of remote collaborations. DAP concentrates on the interactive collaboration but has only limited capabilities for progress tracking during the iteration. DAP is more primarily used for conducting the real-time planning meetings during which interactive collaborations are intensively observed. When we evaluated DAP during distributed planning meetings, we noticed a substantial change in interactions between participants at each site: most of the time, they looked at the shared screen instead of looking at each other. We believe that this is the result of putting a PC projector into the room and not a result of DAP. I.e. all other tools will suffer the same effect. Having a screen at the front of the room and assigning a single person in that room to control the mouse and keyboard, changes the social interactions between team members. When we observed DAP-based planning



Fig. 4 Writing a digital card and the scenario of distributed teams sharing an APDT interface for their agile planning

meetings, participants seemed to be less engaged in the planning process than in traditional agile planning meetings.

3.6 Digital Tabletop-based Agile Planning Tool

Digital tabletops [10] are novel user interaction devices. It has a horizontal display (a table that IS a computer display) and a multi-touch enabled surface to support concurrent touch-based interactions with that display. Agile Planner for Digital Tabletop (APDT) explores using digital tabletops for supporting distributed agile planning. It employs the interactive features of tabletops to enhance the user experience during distributed agile planning meetings. Using touch-sensitive interfaces and a handwriting recognition engine, APDT implements handwriting functions to simulate writing on a paper-based story card. As the tabletop has a horizontal and tangible screen, participants can sit or stand around table, using stylus or fingers to touch the virtual cards on the table surface (see Figure 4). The multi-touch capability enables APDT users to concurrently interact with story cards without being hindered by each other. Telepointers are used to display touch interactions from remote sites.

An advantage of APDT over project planning tools using vertical displays/PC-projectors is that it simulates the co-located project planning and supports several user behaviors that were lost by using the traditional PC-based tools. APDT allows for using pens to write story cards, passing cards on the table surface, using a finger for dragging a card to a participant's territory and reorienting cards. The limitation of using APDT is that present tabletops are not widely available in industry. Only a small number of tabletops is commercially available (e.g. Microsoft Surface and SMART Table) and the purchase price is substantially higher than a PC projector. However, we believe that tabletops will become available in many industrial settings in the future.

Criteria	DAR	CardMeeting	AbloPlanner	MASE	Rally	VelocityOne	XPPlanner	Scrumworks	GitWiki	APPT	Mingle	IBM Jazz
Creating, Editing and Deleting planning objects	F	F	F	F	F	F	F	F	F	F	F	F
Handle efforts estimates	F	N	F	F	F	F	F	F	N	N	F	F
Planning multiple iterations	F	P	F	F	F	F	F	P	P	F	F	F
Moving stories from one iteration to another	F	P	F	F	F	F	F	P	P	F	F	F
Authentication	N	F	F	F	F	F	F	F	N	N	N	N
Real-time updates of the plan	F	F	F	F	F	F	F	F	F	F	F	N
Visual characteristics for different types of stories	F	F	F	F	F	F	N	N	F	F	F	F
Integration with the development environment	F	N	N	N	N	N	N	N	N	N	N	F
Fluid transition between individual and collaborative work	P	P	F	P	P	P	P	P	F	P	P	P
Telepointers for pointing and gesturing	F	N	N	N	N	N	N	N	N	F	N	N
Real time information sharing	P	P	P	N	N	N	N	N	N	P	N	N
Change notification	P	P	N	P	P	?	P	N	P	P	P	P
Joining and leaving meetings	F	F	F	F	F	F	F	F	F	F	F	F
Fluid subgroup formation and dissolution	P	P	P	N	N	N	N	N	P	P	P	N
Simultaneous interaction	F	F	P	N	N	N	N	N	N	F	P	P

Fig. 5 Evaluation table of distributed agile planning tools

4 Tool Evaluation

We now will evaluate agile planning tools based on the requirements for distributed agile planning identified above. The results are shown in Figure 5. In this figure, **F** means the feature is fully supported, **N** is not yet supported and **P** is partly supported, **?** is not enough data collected.

This evaluation table indicates that most requirements of agile project planning are supported by existing tools. Therefore, distributed agile teams can find an appropriate application for project management. However, the requirements for collaborative interactions are not yet or only partly fulfilled. Particularly, the key factors for synchronous interactions: *change notifications* and *telepointers* are rarely supported. The result of this tool analysis matched the results of our interviews with industrial developers. At present, several distributed agile teams are still using Microsoft NetMeeting and Skype for their agile planning meetings and no particular agile planning tools are utilized in the process.

We also conducted a survey to 54 project planning tools published on UserStories website [14]. We concentrate on knowing their application types (Browse, Native, Plugin), the license type (commercial, free), and major functions. The survey shows that:

- **application type:** 44 tools are designed for browser. 3 tools run as a native standalone system, and one tool is plugin to IDE. The remaining 6 tools provide multiple versions to support both plugin and native types.
- **license type:** 11 tools are free product, 22 are commercial product. The remaining 21 tools provide both commercial and free licenses.
- **supporting agile methods:** 15 tools are specifically designed for Scrum and 5 tools are for XP, while 10 tools provide general supports to both XP and Scrum development. The remaining 24 tools do not specify their supporting methods.

- **functionalities:** Only one tool supports synchronous planning, while others provide asynchronous planning features. The features include generating burn down charts, using board (Kanban, Scrum or story boards) to display planning data, strategy supports, timebox management, wiki publishing and agile management logs.
- **supporting multi-language:** Only one tool provides multi-language versions.

5 Practical Advice

In the 10 years of developing and working with agile planning tools along with the evaluation presented above, we present some advice for users and designers of distributed agile planning tools. In addition we discuss our assumptions on the future distributed project planning tools.

5.1 Advice for Agile Planning Tool User

Distributed project planning tools are well developed for supporting asynchronous agile management. The diversity of application types, license types, and their functionalities, provide enough flexibility to choose appropriate tools for supporting specific requirements for asynchronous usage. By supporting, amongst others, Scrum, XP and Lean, agile project planning tools are not restricted to a specific agile methodology. However, when using existing agile planning tools, the following restrictions exist:

- For global agile development, multi-culture and languages are not supported. English is a dominating language for choosing distributed agile planning tools.
- Data exchange between different agile planning tools is problematic. Although existing agile planning tools conceptual support similar artifacts, it is difficult to exchange data between them. Migrating planning data between tools is time consuming and teams need to agree on using a single tool.
- The data exchanging issue also exists between communicating agile planning tools and some general project management applications. We observed some distributed teams using Microsoft Project to manage their development. However, only 2 out of 54 tools listed on UserStories website[14] are able to communicate with MS Project. Within the teams using MS Project and other agile planning tools, data exchanging are still conducted manually.
- Synchronous agile planning meetings are hardly maintained by existing tools. Despite some attempts, such as DAP and APDT were made at employing groupware and digital tabletop technologies to set up synchronous agile planning, none of the tools are fully commercialized and widely applied to industry. Some industrial agile teams are using non-agile specific groupware tools to address the problem. We found one team (distributed over two sites) using desktop sharing

tools to set up a shared environment for agile planning meetings. While this is a pragmatic solution, it limits concurrent interactions and requires a single user at each site to interact with the planning workspace. Looking at long-term developments, we believe the support for synchronous agile planning meetings is becoming a trend with tool suppliers. We expect an increasing number of related tools to become available in the future.

5.2 Advice for Designers of Distributed Agile Planning Tools

Although most existing tools provide enough flexibility and functionality to support progress tracking, the collaborative interactivity needed for distributed agile planning meetings are insufficiently considered. As a result, the experience of industrial in using agile planning tools can - and should - be enhanced. Moreover, the inability of sharing planning data between different tools will be problematic in the future. To better help distributed agile teams, we suggest that the following aspects will improve the usefulness and usability of existing agile planning tools.

- **Supporting synchronous interactions.** As present, synchronous interactions are not well supported by agile planning tools. Designing a practical synchronous interactive system needs to incorporate results from groupware research. Now that support for distributed project management has become more ubiquitous, suppliers need to distinguish their tools by properly supporting synchronous planning meetings. In addition, to enhance collaborative interactions, some advanced technologies need to be incorporated. It is highly possible that the next generation of agile planning tools might have to abandon the PC-projector displays and integrate with new interactive devices like digital tabletops. With the evolution of Web technology, Web browsers will soon support near real-time interactions and the accessibility of synchronous agile planning tools will be improved. The following factors should be considered when implementing synchronous tools for distributed agile planning:
 1. Verbal communication. Tools need to incorporate audio and/or video conferencing capabilities.
 2. A shared card-centered interface: A shared workspace is required for showing detailed aspects of cards, such as card colors, data on the card, and the card positions (considering teams often sort card to show their priorities).
 3. Showing the interactions of remote participants. One of the goals of distributed planning tools is to enhance the collaboration across multiple sites. However, existing tools do not yet show the remote participants' interactions. Telepointers are an appropriate approach to show distributed user interactions. By using remote mouse pointers to monitor the users' interactions (clicking, dragging), the telepointer will show who is interacting with the workspace, and what they are doing. Telepointers also help with identifying the focus of discussion across remote sites. Digital tabletops can show arm shadows (the

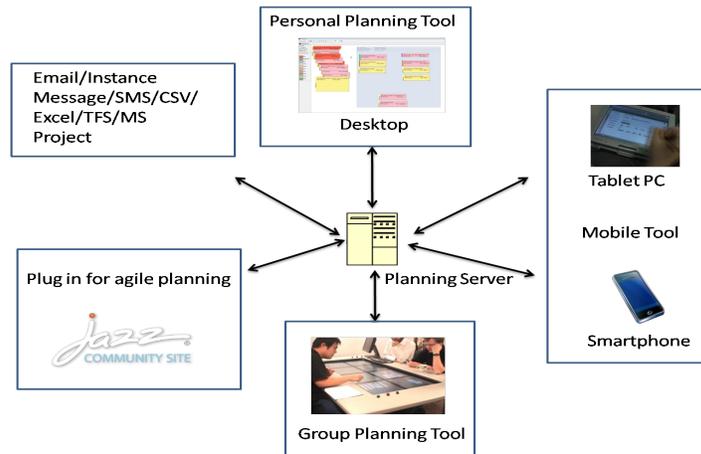


Fig. 6 Ubiquitous project planning model

shadows of users' arms on the table screens) [7]. Thus, the users can not only see hand movements, but also recognize the hand or arm gestures.

In designing synchronous agile planning tools, one needs to combine and implement the above factors in an appropriate manner. Admittedly, some advanced features (such as showing telepointer, or arm gestures) might be restricted by the hardware or software platforms. However, maintaining a card-centered, real-time distributed workspace as well as an audio/video communication are required to support distributed agile planning.

- **Ubiquitous project planning.** Agile project planning often includes multiple roles, such as developers and managers. The diversity of participant raised several types of requirements for having access to agile planning. For example, project managers would like to read the project process reports or burn down charts on their mobile devices. Software developers highlight the use of project planning plugin for their IDE. Meanwhile, both of them would like to have a convenient environment when sitting together to communicate with another distributed sub-teams. Generally, everyone wants to view the project progress from their personal computing environment. In Figure 6 we proposed a model to serve agile planning participants at different environments. Parts of this model, such as plugins and personal planning tools have been implemented. Other components, such as the tabletop based planning tools are still being developed or evaluated. However, a challenging issue for implementing the model is how to exchange data between the various tools. To solve this issue, the following requirement becomes necessary.
- **Exchanging planning data among different tools.** Although agile project planning tools are essentially representing very similar information, none of them can easily exchange planning data with each other. Current tools are closed and create supplier lock in. To solve this issue, we explored the feasibility of exchanging

planning data among different tools. We found most agile planning tools were based on similar conceptual models. A simple translator should be able to bridge the terminology differences among existing tools. We modified a DAP server and added a gateway to translate the planning data from DAP to IBM Jazz and from DAP to the Rally tool. We believe that the integration of different tools will become increasingly important as multiple teams will have to collaborate when agile projects are scaled up. Moreover, card display, editing and management is not enough to support the complete process of agile development. Bug tracing, version control and test automation also play a significant role. Thoughtworks Studio [13] integrates card-based project planning tools (Mingle) with its own release management (version control) products (Cruise), and automation testing platform (Twist). The similar idea is also implemented by the new Microsoft VSTS 2010. Admittedly, it is not easy for most developers, particularly those individual developers developing a complete software package that covers from card planning to release management. However, in designing an agile planning tool, it is beneficial to reserve some extension points (such as data structures that keep the path of one or multiple source code files) to allow source control tools or testing platforms binding their code segments, version updates or testing cases with digital index cards in the agile planning tool.

6 Conclusion

Project management tools for distributed agile teams are currently widely available. They have shown some benefit to supporting project management and knowledge sharing. However, our analysis of existing tools shows that nearly all of them focus on asynchronous features such as supporting progress tracking and card management. The next major step in distributed agile planning tool development will require: supporting synchronous project planning meetings, setting up ubiquitous project planning environments and enabling data exchange between different agile tools and/or with non-agile project planning. These enhancements still require substantial research and development combining agile software development expertise with knowledge about computer supported work and groupware.

References

1. R.Abrahansson, O.Salo, J. Ronkainen and J.Warsta (2002) Agile software development methods, VTT Publications, 112
2. C.A. Ellis, S.J.Gibbs and G.Rein (1991) Groupware: some issues and experiences. Communications of ACM 34(1):39-58
3. C.Larman (2004) Agile & iterative development - a managers's guide, Addison-Wesley, Boston,25-34
4. Jazz Overview (2009) IBM Jazz. <http://jazz.net/> Cited 13 Nov 2009

5. Jeff Dyck, Carl Gutwin, Sriram Subramanian and Christopher Fedak (2004) High-performance telepointers. In: Proceedings of the 2004 ACM conference on Computer supported cooperative work, Chicago, US, ACM, 6-10 November 2004
6. Jonathan Grudin (1994) Groupware and social dynamics: eight challenges for developers. *Communications of ACM* 37(1):92-105
7. Peter Robinson, Philip Tuddenham (2007) Distributed tabletops: supporting remote and mixed-presence tabletop collaboration. In: Proceeding of 2nd workshop on horizontal interactive human-computer systems, Newport, US, 10-12 October 2007
8. Rally Tool (2010): http://www.rallydev.com/agile_products/agile_planning/. Cited 13 Jan. 2010
9. Robert Morgan, Frank Maurer (2008) An observational study of a distributed card based planning environment. In: Proceeding of the 9th International Conference on Agile Processes and eXtreme Programming in Software Engineering, Limerick, Ireland, Springer, 10-14 June 2008
10. S.D.Scott, S. Carpendale (2006) Interacting with digital tabletops. *IEEE computer graphics & applications* 26(5):24-27
11. Thomas Chau, Frank Maurer (2005) A case study of wiki-based experience repository at a medium-sized software company. In: Proceedings of the 3rd international conference on knowledge capture, Banff, Canada, 2005
12. Thomas Chau, Frank Maurer (2004) Tool support for inter-team learning in agile software organizations. In: Proceeding of the workshop on learning software organization, Banff, Canada, 20-21 June 2004
13. Thoughtworks studio (2010): <http://www.thoughtworks-studios.com/>. Cited 13 Jan 2010
14. User Stories (2009): <http://www.userstories.com/products>. Cited 13 Nov 2009
15. Xin Wang, Frank Maurer (2008) Tabletop AgilePlanner: a tabletop-based project planning tool for agile software development teams. In: Proceeding of the 3rd symposium of tabletop and interactive surface, Amsterdam, 1-3 October 2008

About The Authors

Xin Wang is a software developer for the telephony product at Ivnet Inc. He has written and presented on topics such as using digital tabletops to support agile project planning, the design and implementation experiences on tabletop applications and migrating user interface from desktops to digital tabletops. He received his Msc. in computer science from the University of Calgary in 2009.

Dr. Frank Maurer is a Full Professor at the University of Calgary and the head of the Agile Software Engineering (ase) group at the University of Calgary. His research interests are agile software methodologies, engineering digital table applications, executable acceptance test driven development, integrating agile methods and interaction design, framework and API usability, tools for agile teams, specifically for globally distributed software development and experience and knowledge management.

More information about his research can be found at <http://ase.cpsc.ucalgary.ca/>. Currently, the group focuses on empirical investigations of agile techniques, agile product lines, agile interaction design, software design guidelines and application engineering for digital surfaces. He is a member of the Agile Alliance, a Certified Scrum Master, a founding member of the Canadian Agile Network (CAN) - Le

Rseau Agile Canadien (RAC), part of the organizers of the Calgary Agile Methods Users Group and Associate Editor of IEEE Software responsible for the Process and Practices area.

Robert Morgan received a M.Sc. from the University of Calgary's Department of Computer Science and is a former member of the Agile Software Engineering group. In addition to developing distributed agile planning tools, he has had experience working in the financial and oil and gas sectors as a business analyst, developer and agile champion. He is the founder and CEO of Red Duck Solutions, a Calgary based agile software development and consulting firm. You can visit the web site at: www.redducksolutions.com.

Josyleuda Oliveira is a Ph.D. student in Computer Science at the University of Calgary. Her research interest is agile software methodologies, specifically in globally distributed software development. She received a M.Sc. in Software Engineering from University of Fortaleza-Brazil in 2006. She has had 11 years of experience in industry, in Brazil. She worked as a Project Manager, Software Quality Assurance Analyst and Business Analyst.