The dBoard: A Digital Scrum Board for Distributed Software Development

Conference Paper · November 2015

DOI: 10.1145/2817721.2817746

5 authors, including:

Morten Esbensen
IT University of Copenhagen

Paolo Tell
IT University of Copenhagen

Jacob Benjamin Cholewa
IT University of Copenhagen

Jakob E. Bardram
Technical University of Denmark

The dBoard: a Digital Scrum Board for Distributed Software Development

Morten Esbensen, Paolo Tell, Jacob B. Cholewa, Mathias K. Pedersen & Jakob Bardram
The Pervasive Interaction Technology Laboratory
IT University of Copenhagen, Rued Langgaards Vej 7, 2300 Copenhagen S, DK
{mortenq,pate,jbec,mkin,bardram}@itu.dk

ABSTRACT
In this paper we present the dBoard—a digital Scrum Board for distributed Agile software development teams. The dBoard is designed as a ‘virtual window’ between two Scrum team spaces. It connects two locations with live video and audio, which is overlaid with a synchronized and interactive digital Scrum board, and it adapts the fidelity of the video/audio to the presence of people in front of it. The dBoard is designed to work (i) as a passive information radiator from which it is easy to get an overview of the status of work, (ii) as a media space providing awareness about the presence of remote co-workers, and (iii) as an active meeting support tool. The paper presents a case study of distributed Scrum in a large software company that motivates the design of the dBoard, and details the design and technical implementation of the dBoard. The paper also reports on an initial user study, which shows that users found the dBoard both useful and easy to use. Based on this work, we suggest that superimposing collaborative applications onto live video is a useful way of designing collaborative meeting and awareness systems.

Author Keywords
dBoard; Scrum Board; Videoconferencing; Scrum; Task Board

ACM Classification Keywords
H.5.2 Information Interfaces and Presentation: User Interfaces

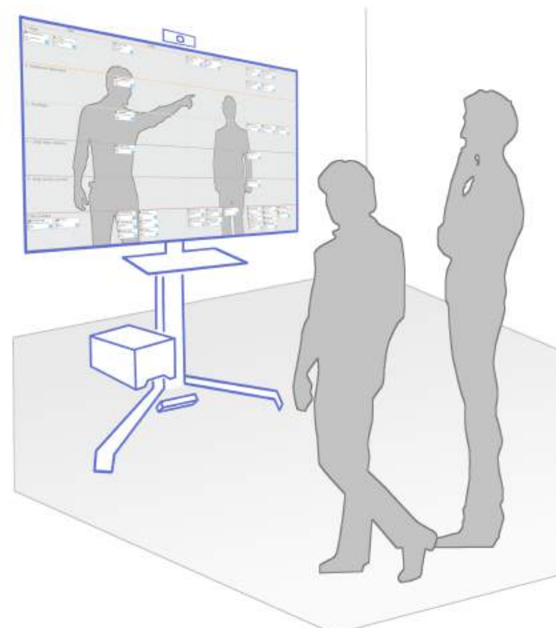

Figure 1. The dBoard is an interactive scrum board and videoconferencing tool designed for distributed teams.

INTRODUCTION
Videoconferencing technologies play a core role in today’s globalized business world, and many types of videosystems are used from low-cost and easily available technologies like Skype and Google Hangout, to high-cost dedicated videoconferencing rooms like the Cisco Immersive TelePrecense system. However, when people meet using video it is often with a specific task at hand. Studies have shown that videoconferencing is often conducted as part of a shared experience [4], and that there is a need for integrating videoconferencing with other systems that are used in such arrangements [25]. Moreover, most videoconferencing systems are often only designed for planned meetings in dedicated meeting rooms or spaces, and therefore do not support ongoing awareness information that comes from seeing and informally interacting with co-workers in a shared office space [1].

One important application area for videoconferencing is distributed software development. In particular, in Agile development methods, like Scrum, which puts an emphasis on close collaboration and frequent meetings, videoconferencing has become essential in coordinating work across remote sites [18]. Core to Agile software processes is the use of a Scrum board, which is a large physical task board situated in the software development team’s office [19]. The board works as a boundary object, helping to reduce the amount of articulation work [20] and serves two main functions: during the daily meetings it is used to guide the discussion on task progression, and during ordinary work it is used as a passive information radiator showing the status of work as team members update the information on the board. However, the benefit of the Scrum boards relies on the fact that the team is collocated with the board, and a (physical) Scrum board is

ITS 2015, November 15–18, 2015, Funchal/Madeira, Portugal.

ACM 978-1-4503-3899-8/15/11 ...$15.00.
DOI: http://dx.doi.org/10.1145/2817721.2817746

less useful for distributed software development. While concerns have been raised with respect to replacing physical task boards with digital counterparts [23], recent studies suggests that companies will adopt and use technological solutions as their Scrum board [2, 11].

In this paper, we present the design, implementation and evaluation of the dBoard —a digital Scrum board designed specifically for distributed Scrum teams. The dBoard as conceptualized in Figure 1 is designed to support three main functions: (i) as a regular Scrum board revealing task information; (ii) as a media space providing awareness about the presence of remote co-workers, and (iii) as an active meeting support tool during daily Scrum meetings. Instead of separating support for videoconferencing and Scrum board, the dBoard integrates these two by superimposing the Scrum board onto a full screen video-channel. The design of dBoard was based on insights from an observational study of a distributed Scrum team. Once the dBoard was designed and implemented, it was subject to a scenario-based evaluation with Scrum practitioners.

This paper contributes to a detailed understanding of the hardware, software, and user interaction design and implementation of 'collaborative window' technology, which is comprised of a full screen video feed overlaid with domain-specific information—in this case designed to support a distributed Scrum team. Moreover, the study of the dBoard revealed that practitioners found it useful and easy to use, arguing that dBoard would reduce the setup cost of engaging into collaborative ad-hoc meetings, while also pointing to areas for improvements to increase its efficiency in work.

RELATED WORK

dBoard builds on and extends prior research on immersive teleconferencing, media spaces and Scrum boards.

Immersive Teleconferencing

The idea of superimposing a shared user interface on top of videoconferencing has been researched in a number of projects. ClearBoard [16] was designed based on a metaphor of looking through and drawing on a transparent window into another office. This line of research have since been extended to include systems that capture the image of a user through half-silvered screens to provide parallax free videoconferencing systems that convey eye contact [17, 24]. Recently, 3D sensors have been used in conjunction with cameras to capture people and provided interfaces in which 3D realistic representations of participants are presented blended with user interface elements [13, 26]. Despite such focus on gaze and gesture, these systems demonstrate an idea in which videoconferencing and task specific user interfaces can be combined into one technology instead of keeping each tool in its own window or on separate screens. This research has mostly focused on techniques for preserving eye-contact and conveying gestures to provide more immersive teleconferencing, and less on how to make the content shown on top of the video interactive.

Media Spaces

Traditional videoconferencing setups have mostly been designed for meetings. However, in collocated environments, much information is conveyed outside meetings by being aware of the presence of nearby colleagues or during informal talks outside scheduled meetings. This fact has been recognized by system designers who have proposed videoconferencing systems capable of providing awareness information. Portholes [7] demonstrated an early version of such systems in which still images were broadcasted throughout an office space to provide co-workers sitting in different offices with awareness of each other. The PARC media space [1] and VideoWindow [9] connect different offices using live video. More recently, systems such as MirrorSpaces [21] and Pêle-Mêle [12] further explored the concept of media spaces by offering always-on video systems that take into account the proximity of people to the system screen as a means of altering the video.

Scrum Board Technologies

While the Scrum board as an artifact has been subjected to different studies (e.g. [11, 23]), less focus has been on developing new tools. Rubart has proposed using an interactive tabletop-based Scrum board as a tool for collocated Scrum [22], and Boden et. al. demonstrated how an implementation of articulation spaces was appropriated as a Scrum board for a collocated software development team [2]. Along similar lines, Agile planning on tabletops has been demonstrated with the APDT system [10]. However, in general research on collaborative digital Scrum boards is scarce and little is known about how to design Scrum boards for distributed teams.

In comparison to prior research, the dBoard contributes an immersive teleconferencing system that supports displaying a highly dense information overlay with detailed Scrum information, while also supporting direct touch-based interaction with this content. The dBoard is designed to work as both as a video meeting tool as well as a media space for remote awareness, and it utilizes a proxemics-based interaction model to seamlessly move between these two modes of operation.

DESIGN

The dBoard is designed as part of a larger project studying the work of distributed software developers[1]. The motivation of designing the dBoard came partly from the related work we described and partly from an ethnographic field study of distributed software developers [8]. The design was done in a user-centered iterative design process involving two different companies in the studies and design process.

Field Study of Distributed Software Development

To understand the challenges of distributed Scrum we conducted a field study of such a collaboration. The collaboration we observed involved a group of developers distributed across India and Denmark. The group was one coherent

[1]The *NexGSD* project is a research project dedicated to studying global software developement.

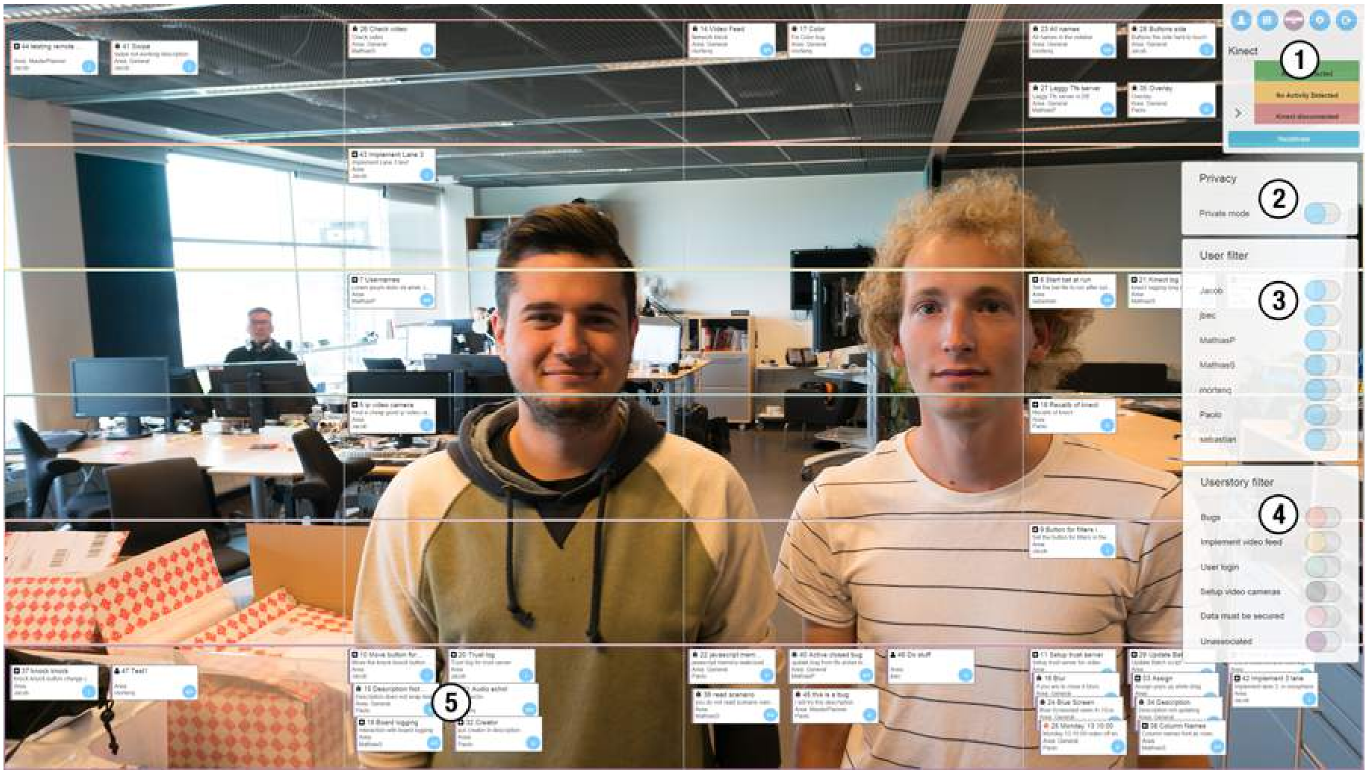

Figure 2. The user interface of the dBoard with menu (1), privacy menu (2), user filters (3), story filters (4), and tasks (5)

Scrum team of ten people working on a financial product for the Scandinavian market. Two researchers—one located in Denmark, one located in India—observed the collaboration concurrently. The study applied participant observations in which we observed daily work and meetings of the developers and onsite interviews with team members.

Our observations identified a set of recurring challenges in running an Agile distributed software development process. First, the Agile methodology encourages team members to do a short and efficient daily stand-up meeting for easy and low-level coordination. Such meeting should take place in the shared workspace of the team in front of the Scrum board and should not exceed more than 15 minutes. However, because the team was distributed, setting up and running daily stand-up meetings was cumbersome. The problems often were related to the lack of proper tool support in the right location—both for video meetings and for handling tasks on a Scrum board. For example, the team distributed between Denmark and India would try to meet at the same time each day (9:45 CET in Denmark and 1:15 PM in India). For this to happen, the team first had to go to dedicated videoconferencing rooms at each site. Then, they had to set up the videoconference call and all the software development tools needed. The meetings were usually initiated by an Indian developer setting up all the required systems and then calling the Danish side. The team used a traditional videoconferencing setup in conjunction with the HP Application Lifecycle Manager (ALM) system to manage and update their tasks. As this application was not designed for collaborative use, the Scrum master—located in Denmark—controlled the ALM application while the team in India used VPN and Remote Desktop Connection to connect to the Scrum master's computer and project the application onto a screen next to the videoconferencing screen. As such, the amount of work needed to set up and conduct a stand-up meeting was extremely demanding, and, when the experienced developer was not around to start the meeting, we observed how the others struggled to get the video conferencing, ALM, VPN and Remote Desktop Connection up and running. Hence, these meetings were far from the ideal Agile stand-up meeting taking place in the developers' workspace and only lasting 15 minutes.

A second recurring challenge we observed in the distributed teams was the lack of mutual awareness, especially outside the daily stand-up meetings. In one instance for example, two Indian developers were working on the same bug. When using a traditional Scrum board in a collocated team workspace, such a situation would seldom happen, since a developer would physically go to the board and move the 'bug ticket' from one column to another, thereby signaling a status change. However, in the distributed setup there was no way for a developer to notice such a change if not explicitly communicated. To account for this lack of awareness between the developers, the lead developer located in Denmark contacted the Indian developers on a daily basis using instant messaging. This process allowed him to get an overview of the status of work, but this information then only resided with him and no mutual awareness among team members was maintained.

SYSTEM OVERVIEW

The dBoard is an augmented version of the traditional physical Scrum board that integrates video and Scrum board features into one single tool designed to run on a large multi-touch surface. Based on the field study we formulated the following set of core design features that was used to design the dBoard:

- *Virtual Window*—The dBoard should provide a 'virtual window' between two remote sites thereby enabling seamless communication between two distributed teams.
- *Scrum Board*—The dBoard should support task management (i.e., user stories) in a Scrum fashion by supporting direct manipulation of 'virtual post-it notes' on a grid-like task board.
- *Proximity-based Interaction*—The dBoard should adapt to the environment in which it is situated; in particular, it should adapt to whether people are in front of it or not and to how they are using it.
- *Minimal Setup Costs*—The dBoard should allow people to easily establish meetings with minimal setup costs.
- *Tool Integration*—The dBoard should integrate to relevant digital software engineering tools like issue tracking and Agile project management used in the Scrum practices.

We designed and implemented a prototype based on these design guideliens and presented this prototype in a workshop with a company that works with distributed Scrum. Based on the feedback from this workshop we finalized design and implementation of the dBoard. In the following we describe the features of the dBoard in detail.

Video Window

The dBoard applies a 'window' metaphor to its video conferencing features. The whole background of the dBoard user interface is a large video stream from a connected dBoard designed to give the same feeling that can be experienced when looking through a window into another office. All other user interface elements such as the Scrum board and the menus are placed on top of this video. Similar to a media space [1], the dBoard video is always on and is started automatically when the system is started. The dBoard also captures and streams audio from a connected microphone. This always-on feature makes initiating a meeting as simple as walking up to a board. To get the attention of people at the location of another dBoard, we implemented a 'knock knock' feature. When a user performs a knocking gesture on the board, a knocking sound is played at the other board signaling that someone requests attention.

Scrum Board

The Scrum board elements of the dBoard are superimposed onto the full screen video stream from the other dBoard (see Figure 2). The Scrum board is organized as a traditional physical board: tasks are represented as small digital post-it notes that are arranged into columns indicating their state and rows representing the user story they belong to. Tasks not belonging to a user story are placed in an *Unassociated* row. The tasks display information about their name, type, the user story they belong to, description and the person assigned to it as seen in Figure 3. Tasks can be repositioned using touch,

Figure 3. A closeup of a task on the dBoard with task color (1), task type (2), title (3), description (4), assigned user (5), assign button + user initial (6), and assign dropdown (7)

and dragging a task from one column to another changes its state. Tasks can be placed anywhere on the board and do not 'snap' into position when dropped allowing users to arrange the tasks on the board in a similar way that post-it notes can be arranged on a physical board. To assign a task to a developer, users can tap the task's name or user icon to access a dropdown list of people to whom the task can be assigned (see Figure 3).

The state of the dBoard is kept synchronized across the two connected boards and all task movements on one board are immediately updated on the other. As in traditional video-conferencing setups, the camera on the dBoard is mounted on top of the stand, which makes it impossible to point to specific tasks due to the parallax. To address this issue we implemented interaction awareness on the screen. When a user drags a task, the task is highlighted in red on both boards. When a user touches anywhere else on the board, a small red pointer is shown on both boards.

We also implemented a number of options to filter and sort the tasks on the board. Two filter menus (Figure 2(3 and 4)) can be used to highlight specific tasks based on two parameters: user and user story. When a user selects a user or user story filter, the dBoard opacifies all tasks that do not match the given filter. As tasks can positioned anywhere on the board we also implemented two ways to sort the board. When sorting, tasks are automatically positioned in the row and column that correspond to their state and user story. To sort the entire board, a user can tap the sorting button in the menu. To sort only a specific tile (i.e. row/column intersection), a user can tap and hold that tile. As the state of the board is kept synchronized across the two connected sites, sorting in one end results in tasks on the other board being sorted as well. Lastly, the rows on the dBoard can be switched to show users rather than user stories. This means that all tasks are re-positioned to be located in the row representing the user they belong to.

Tasks on the dBoard are deliberately designed to be small post-it notes-like elements (Figure 3). These notes contain the very basic information about the task however, to access detailed information users can bring up a side panel. The panel (Figure 4) slides in from the right side of the board when a use performs a tap-and-hold gesture on a task or a user story and it contains much more detailed information about the task. The side panel contains task name, full description, state information and legal states, time estimate and change history list. From the side panel, a user can change the time estimate by performing a dragging gesture on the progress bar or change the state by tapping one of the state buttons. The side panel can be hidden again by performing a swipe gesture on it.

Proxemic Interaction

To provide a seamless transition between awareness tool and meeting tool, the video of the dBoard is automatically blurred and de-blurred and the audio turned on and off by sensing the proximity of people. If no people are in front of the dBoards at both ends, the video is blurred and the audio is turned off. If people are present at only one of the dBoards, the video is blurred less in both ends while the audio is kept turned off. If people are present in both ends, the video is completely un-blurred and the audio is turned on. The blur is used to enhance visibility of the Scrum board – when blurred, the contours in the video are washed out and the colors are faded thus bringing forward the non-blurred user interface elements of the scrum board. Furthermore, the audio is only turned on if people are present at both ends to ensure that the dBoard does not become too much of a distraction if placed in a busy working environment. While blur has been shown to be able to provide the balance between awareness and privacy [3], we also implemented a privacy button that instead of blurring the video removes it altogether until people are present at both boards. Figure 5 shows a example these three modes.

To provide transparency to users with respect to what the dBoard senses, the proximity button in the menu changes color based on the state of the sensing (Figure 2(1)). When green, the button signals that the dBoard has sensed one or more people in front of the board, yellow signals that the sensing is running but no people are seen, while flashing red signals a disconnection to the sensing application. This information is also available when tapping the button in which case a dropdown appears in which the colors are explained in more detail and the current state is highlighted.

Minimal Setup Costs

We built a prototype of the dBoard mounted on a wheeled stand that can easily be moved anywhere. Alternatively, the screen of the dBoard can be setup using a wall display. We thus want to stress that the hardware of the dBoard system can be setup in many configurations as it doesn't require any specially engineered hardware or large physical space. This allows the dBoard to be deployed in the place that best fits its needs or that would be chosen for placing a physical board—for example among the developers. When the system is started, the dBoard application is automatically launched and the video and audio connection established.

Tool Integration

All sprints, user stories, tasks, bugs and users are synchronized with a Microsoft Team Foundation Server (TFS) installation. The synchronization makes sure that all information on the dBoard is taken from a TFS instance and that all changes made either on the dBoard or on another system are

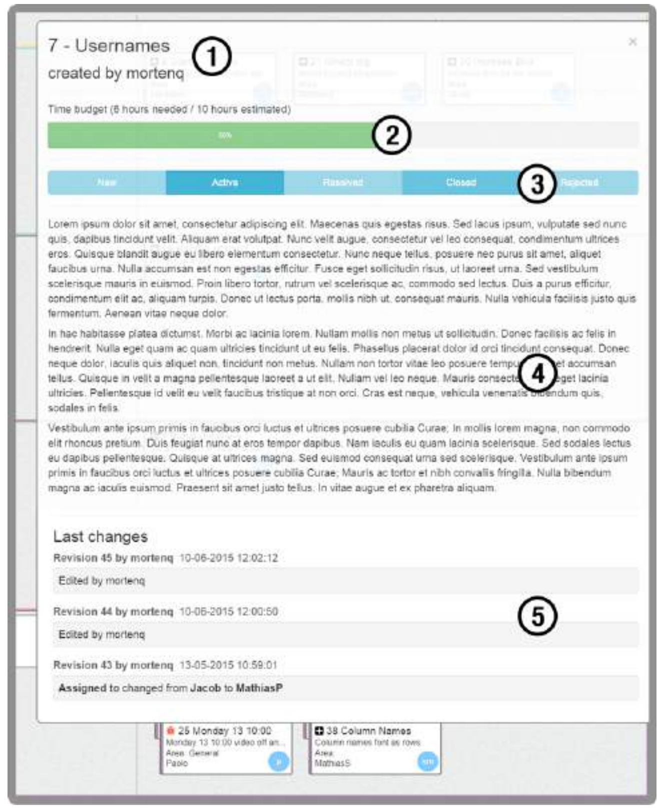

Figure 4. The side panel of dBoard can be used to see detailed information about tasks and user stories. The panel contains title (1), time estimate (2), state (3), rich text description (4), and change history (5)

kept synchronized across the dBoards and TFS. In case of inconsistency between the dBoard and TFS due to network problems, TFS is considered the master.

IMPLEMENTATION

The dBoard system consists of two main parts: the *dBoard*, which is the board itself; and the *backend* which handles communication between dBoards and integration with external tools. This section presents the implementation details of these parts.

dBoard

The dBoard hardware consists of a large 65" display with a 3840 x 2160 resolution and PQ Labs multitouch overlay. A connected high-definition camera and a quality microphone

Figure 5. The dBoard in its normal mode (left), with blurred video (middle, and with privacy enabled (right).

captures video and audio. All the hardware is mounted on a movable stand and all components are connected to a fan-less computer strapped to the stand. The setup ensures that setting up and starting the dBoard is as simple as moving it to the desired location and plugging in power. The Kinect v2 sensor used to sense proximity to the board is mounted between the wheels of the stand. Alternatively, the screen can be hanged on a wall.

On startup, the dBoard launches the dBoard frontend application in fullscreen. The dBoard frontend is an HTML5 and JavaScript application built with AngularJS and designed for Google Chrome. Two connected dBoards send and receive video and audio using WebRTC, which is a peer-to-peer based JavaScript framework while all other information (such as tasks, user stories and state information) is communicated through the server using websockets.

Proximity Sensing

The dBoard senses proximity with a Microsoft Kinect v2 attached between the wheels of the system. The Kinect is accessed through the official .NET SDK in a C# application, which analyzes skeletal information and depth frames to infer human presence. Depth frames are used in addition to skeletal tracking as it is possible for people to stand too close to the dBoard for the Kinect to pick up skeletal information. The depth frames are analyzed in two ways: large frame-by-frame pixel changes suggest movement in front of the sensor and a large summed difference between the current frame and a calibration frame suggests proximity. In case presence information is sensed for an extended period of time without any movements happening, the application assumes that a non-human artifact (such as a chair) has been placed in view of the camera and the application recalibrates automatically. A menu button also allows to manually request a recalibration of the proximity sensing application. All presence information is sent to the dBoard web application over a websocket.

Backend

As shown in Figure 6, the dBoard backend consists of three main components: (i) a Node.js web server, (ii) a Microsoft Team Foundation Server (TFS), and (iii) the activity-centered infrastructure NooSphere [15] with a plugin to communicate with the TFS instance.

The Node.js server is responsible for serving the AngularJS web application to the dBoards along with facilitating websocket connections for task and interface synchronization between the boards and the backend. Any change made on the dBoard or in TFS is synchronized through NooSphere. NooSphere exposes all Scrum activities (i.e. sprints, user stories, tasks, bugs and users) through a RESTful web interface. We added this extra layer on top of TFS as it allows for easily changing the task tracking software from TFS to, e.g., Jira without modifying the rest of the application. Changing or adding task management systems can be done by writing a plugin. Other tools would then in turn also be able to access the same data in NooSphere making it an infrastructure able to support other collaborative tools.

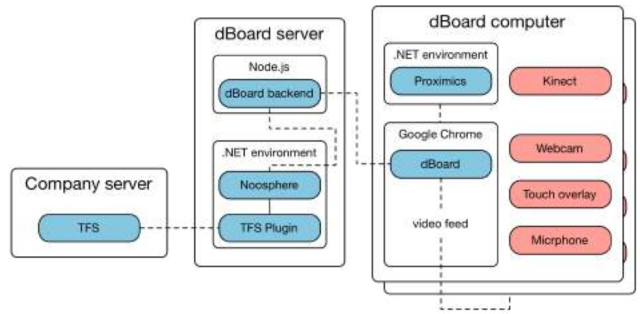

Figure 6. An overview of the architecture of dBoard with its three main components; the TFS server, the dBoard server and the dBoards.

STUDY

To evaluate the design of the dBoard, a scenario-based user evaluation [5] was done. The goal was to (i) study perceived usefulness and ease of use, (ii) observe how participants would use dBoard for performing standup and ad-hoc meetings, and (iii) gather feedback on the features of dBoard.

Participants

The study was conducted by inviting local software development companies using Scrum to participate in an evaluation workshop. Seven software engineers from three different companies participated in the study (age μ = 30,5; σ = 4,53). Participants reported to be experienced Agile and Scrum practitioners, their seniority ranging from 6 months to 10 years (μ = 4,3; σ = 3,98). wo participants were Scrum masters, two product owners, and the rest were developers or testers.

Method

Figure 7 shows the evaluation setup. One board was placed on the stand shown in Figure 1 while the other board was deployed on a large wall display. This setup enabled us to evaluate different hardware form factors of the dBoard system.

The workshop was divided into four parts: (i) an introduction to the dBoard in which the participants were welcomed to the study, signed an informed consent form, filled out a demographic questionnaire, and were given a detailed presentation of the dBoard features; (ii) a hands-on session in which the participants were invited to experiment with the two different dBoards; (iii) a session in which we asked participants to act in two different scenarios; and, (iv) a closing group discussion in which participants, after completing a short survey, were asked to reflect on the dBoard and elaborate on their survey answers.

The two scenarios were chosen from common activities in traditional distributed Scrum teams. Scenario I required the participants to first perform a simple design task (i.e., select a template for a website design based on inspirations from existing websites); update the status of the task on the dBoard; and, contact the remote team (enacted by confederates [5]) to communicate that the design task was completed as well as seek information about the progress on the remote site. Scenario II simulated a standup meeting; participants were given

Figure 7. Snapshot of the evaluation taken during Scenario II—the 'standup meeting' using the dBoard on a mount (left) and on a wall-sized display (right).

a script describing the role of the enacted character (one of which was the Scrum master dictating the pace of the meeting), the work done on the previous day, the one planned for the coming day, and whether there were some impediments or clarifications required from the remote site.

The survey applied the technology acceptance model (TAM) [6] to which four questions were added to assess the usefulness of specific dBoard features (i.e., videoconferencing, Scrum board, privacy through proxemic, and 'knock knock'). Each question was measured with a seven-point Likert scale.

Results

Figure 8 depicts an overview of the results of the questionnaire on the perceived usefulness, perceived ease of use, and usefulness of the different features of the dBoard.

Perceived Usefulness

Even though overall participants scored positively the usefulness of the dBoard, on average they remained quite neutral with answers that varied across participants. In particular, they reported to be uncertain about the ability of the dBoard to enable them to accomplish tasks more quickly ($\tilde{x} = 4$, iqr = 2); a pattern that is repeated across the next four questions related to the ability of the system to: improve performance ($\tilde{x} = 3$, iqr = 0); improve productivity ($\tilde{x} = 3$, iqr = 1,5); enhance effectiveness on the job ($\tilde{x} = 3$, iqr = 0); and, make it easy to do the job ($\tilde{x} = 3$, iqr = 1,5). Nonetheless, on a more direct question about the usefulness of the dBoard in their job, the participants scored the dBoard very positively ($\tilde{x} = 1$, iqr = 1). When asked about the usefulness of the appication in the post-evaluation interview, several participants mentioned that the dBoard should support sprint planning through access to features of creating new tasks at the board and access to the backlig.

Perceived Ease of Use

All participate managed to appropriate the system with more or less confidence in a very brief period of time and reported that it was easy to become skillful in using the dBoard ($\tilde{x} = 2$, iqr = 1); they reported the dBoard to be very easy to learn ($\tilde{x} = 1$, iqr = 1) and to use ($\tilde{x} = 2$, iqr = 1). The analogy with the post-it notes often used in physical Scrum boards was well received; participants appreciated the flexibility provided by the dBoard in terms of freely arranging tasks at arbitrary locations ($\tilde{x} = 2$, iqr = 1,5). The question related to how clear and understandable the dBoard was however, received some negative scores ($\tilde{x} = 2$, iqr = 1,5); a trend visible also in the question related how easy it was to control the dBoard ($\tilde{x} = 3$, iqr = 2,5). The ease of use was also apparent from our observations of the participants during the evaluation sessions. Even with the limited training and experience of interacting with the dBoard from the evaluation introductions, all participants confidently interacted with the boards to complete the evaluation scenarios.

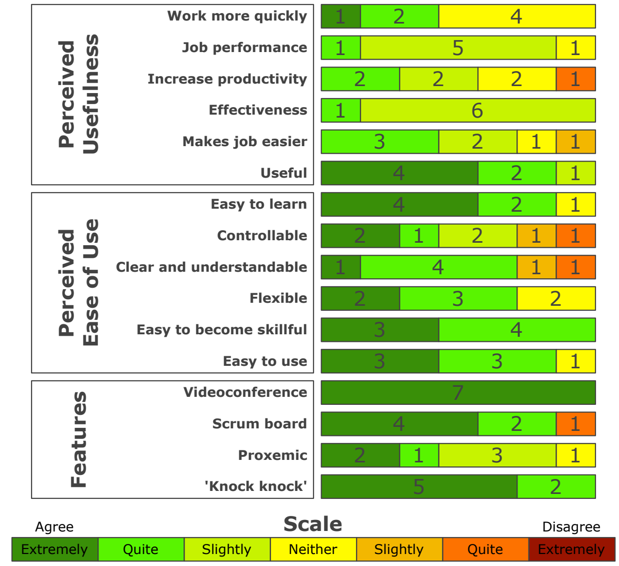

Figure 8. The results of the 7-point Likert scale questionnaire on perceived usefulness, perceived ease of use, and usefulness of the different dBoard features. The figures in each cell shows the number of answers for a given score.

Feature Usefulness

When confronted with more specific questions about the usefulness of the different features provided by the dBoard, participants provided very positive scores. Especially with regards to the videoconferencing capabilities, all participants considered this feature extremely useful (Videoconference: $\tilde{x} = 1$, iqr = 0) and provided comments like:

– *"The entire video is really cool – really helpful."* – *P3* – *"You are actually interacting with your colleagues that you do not normally see."* – *P5*

The Scrum board features were also received very well (Scrum board: $\tilde{x} = 1$, iqr = 1); participants found the integration with TFS to be an essential feature for a digital version of a Scrum board as it represents the main shortcoming of the physical counterpart, and they saw a significant value in the affordances provided by the dedicated setup mimicking physical Scrum boards. One participant stated:

– *"One of the things I like of a Scrum board like this is that you have not the digital but the post-its like interactions, and you can walk up there, take a task up here, and put it over here. I know you can do it on your computer—it might be easier—but I just like that feeling, I like to be able to do that."* – *P5*

Moreover, the simplicity of the scrum board representation was also appreciated; as one of the participants ironically pointed out:

– *"[speaking about a feature] it is definitely also there [in Jira], but it would take a couple of days to find out where." – P7*

In relation to the proxemic interaction that supports gradual engagement and privacy, participants found it less useful (Proxemic: $\tilde{x}$ = 3, iqr = 1,5). On the one hand, some participants valued the ability of mitigating the disturbance of a constant video feed from a large display:

– *"Without the blur, it would be like [being] in a sport bar during a soccer match where everyones' attention is capture by the game..." – P2*,

– *"I like the idea that when one team is working, they can see when a person is at the board." – P4*

On the other hand, the quick and automatic de-blurring of the video feed due to the detection of presence on the remote site was considered by some as potentially distracting. Participants also mentioned that it would be useful for the content to change based on proximity of people. One participant for example, mentioned that the Scrum board could change to a burndown chart.

Finally, the 'knock knock' feature was also found very useful ('Knock knock': $\tilde{x}$ = 1, iqr = 1). One participant stated:

– *"[...] we use Slack for things like 'do you have time for a Skype call?'. And that is basically the 'knock knock' right, so I think you got it pretty good there." – P3*

Some participants also suggested using the knock-knock gesture to initiate de-blurring so this would happen via an explicit gesture rather then through the implicit proxemics-based interaction.

– *"[...] it should only activate [the video] when you knock " – P4*

DISCUSSION

This paper contributes to a detailed understanding of the hardware, software, and user interaction design of 'collaborative window' technology, which is comprised of a full screen video feed overlaid with domain-specific information—in this case designed to support a distributed Scrum team. This section presents a discussion of this design space, the lessons learned from the evaluation study, and the more general concept of a 'collaborative window'.

Hardware, Software, and Interaction Design of dBoard

The design of dBoard shows how an ad-hoc video conferencing tool can be integrated with a Scrum-based task management board. While prior research have shown such system design (e.g. [16, 24, 26]), these systems apply advanced and special-designed hardware setups to remove camera parallax and to correctly convey eye-contact, gaze and gestures. This has not been the focus of dBoard, which instead uses off-the-shelf top-mounted cameras and displays. This does introduce camera parallaxes, but in our evaluation participants did not consider this a problem. Moreover, in contrast to prior work, dBoard does not flip the remote image horizontally. This means that if a participant points to a ticket on the left of the Scrum board, the remote video image will show him pointing to the right and hence not on the right ticket. If the design was to accommodate this problem, the display could be flipped horizontally in two ways: either the video or the content. In the design of dBoard, neither of the two were flipped. The content (i.e., the Scrum board) was not flipped, since tickets flow from the left column ('backlog') to the right column ('done') hence, horizontal orientation carries a semantic meaning. Moreover, if the content was flipped, then all text on the Scrum board (including each task ticket) would be mirrored and hence unreadable. The video was not flipped either, since this would distort the view from the remote site into the local office, and would—we argue—contradict the 'window' design metaphor that allows users to directly look into the remote office however, one participant mentioned in the scenario that the video could be flipped. In addition, participants were able to see from the remote pointer what task was being pointed at and moved around.

The overall interaction design of dBoard incorporates two core design principles, which could be of general use for collaborative technologies; 'reducing startup costs' and 'beyond being there'.

The design principle of *Reducing Startup Costs* addresses the need for making it extremely easy and effortless to engage in collaboration. Rather than having to book and setup a video meeting, dBoard was designed as an 'always-on' video window to the other site. It would always show the current status of the Scrum board and all task information is easily accessible through a very straightforward gesture-based touch interface. As such, there is a very limited cost in terms of time, effort, and cognitive load associated with engaging in a distributed ad-hoc Scrum meeting – the dBoard is literally a 'walk-up-and-use' technology. This was also recognized by the study participants, who argued that this was a core benefit of the system (as e.g. reflected in the high score on the 'videoconference' feature evaluation).

The design principles of *Beyond Being There*[2] addresses how technology can go beyond collocated interaction. As said, Scrum is inherently based on collocated practices where the Scrum team meets, works, discusses, and demos in a team room with a Scrum board. This collocated practice allow for high-bandwidth informal communication but it also carries with it the problems of open-space offices in terms of noise, disturbances, (lack of) privacy, and stress. The dBoard allows a team to reduce (blur) noise and disturbances from the remote team and to preserve local privacy. As noted by P2 during the evaluation; blurring the video made it less disturbing. It must, however, be underlined that the current study of dBoard did not compare collocated with distributed Scrum practices; this is the subject for a future study.

Perceived Usefulness and Usability

In general, the study of dBoard showed that practitioners found it useful and easy to use, arguing that it would reduce the setup cost of engaging into collaborative ad-hoc meetings,

[2] This principle paraphrases Hollan & Stornetta's CHI'92 paper [14].

while also pointing to areas for improvements to increase its efficiency in work.

The evaluation of the usefulness of the dBoard showed some mixed results. On one hand, the *overall* usefulness and the usefulness of the four specific features (videoconferencing, Scrum board, Proxemic, and 'Knock-knock') were rated high. On the other hand, specific usefulness parameters on *performance*, *productivity*, and *effectiveness* received moderate scores. When interviewing participants, they argued that they did not see dBoard as a productivity-enhancing tool, but rather as a convenient way to conduct remote ad-hoc Scrum meetings and to maintain an awareness on the status of work. As such, dBoard would not 'enhance' or 'improve' on anything they did not do already. Moreover, the participants also pointed out that the dBoard had to incorporate support for sprint planning, if it was to be used throughout the entire Scrum process.

In general, the evaluation showed that participants found dBoard easy to use. There are, however, still some performance issues that affect ease of use, which was also reflected in the evaluation scores of the TAM (Controllable and Clear and understandable). During the evaluations, some gestures such as swiping or dragging multiple tickets suffered from 'lagging'. Despite the computational power of today, we experienced that implementing such a complex collaborative application as a web application could cause some performance bottlenecks. Multi-touch and gesture-based interaction on web application is still in its initial stage and needs to be improved before dBoard can get widespread adoption.

The evaluation showed that the specific dBoard features were all found very useful, with the videoconferencing as a significant high score. During the interviews, several issues and suggestions for improvements were raised by the participants. A recurrent question was related to 'information overload', i.e. what happens if too much Scrum information is shown on the dBoard display? Is there a risk of covering most of the video? Another line of comments was related to the usage of proxemics. Several participants mentioned that they would prefer to have the de-blurring of video triggered by the knock-knock feature rather than the proxemic interaction. This later issue relates to whether blurring – or any other feature – should be based on implicit (proxemics) or explicit interaction. This limited study shows that users tend to prefer explicit gestures for initiating (de-blurring) a video meeting, whereas automatic blurring the video once users is no longer in front of dBoard seems to be feasible. However, studying the pros and cons of implicit interaction in dBoard would require a longer deployment study.

Collaborative Windows

Based on our work of designing, implementing and evaluating dBoard, we argue that combining video with collaborative user interfaces using a 'collaborative window' metaphor is a promising design guideline for collaborative applications. A collaborative window is designed as a window into a distant location upon which interactive collaborative content is superimposed. Despite not solving camera parallax or providing support for gaze and gestures, we demonstrated how a collaborative window application such as dBoard was perceived very well by users with respect to usefulness and ease of use. In particular, such applications allow for a richer collaborative experience as they do not require users to constantly switch attention from the video to the tasks at hand and the combination of video and content furthermore reduces startup costs.

CONCLUSION

In this paper we presented the design, implementation and evaluation of the dBoard—a digital distributed Scrum board that blends together videoconferencing and Scrum task management. This is achieved by bringing them together in a setup where the Scrum board specific elements are superimposed onto the video. The dBoard was designed to provide support both as a passive information radiator from which the state of work can be collected, as a media space proving awareness about the presence of remote co-workers, and as an active meeting support tool. By sensing the proximity of people the dBoard provides a way to seamlessly switch between these three modes of operations.

We designed the dBoard based on observations of distributed Scrum practices and implemented the system as a web application that runs on large multitouch enabled screens. The dBoard was evaluated in a scenario based setting with experienced Scrum practitioners. The evaluation revealed that users found the dBoard both useful and easy to use. In particular, the combination of video and Scrum board was very well received. Based on this work, we argue that adopting the metaphor of a 'collaborative window' as a means of designing collaborative videoconferencing systems can lead to new and interesting applications supporting distributed collaboration.

ACKNOWLEDGMENTS

This research has been funded by the Danish Agency for Science, Technology and Innovation under the project "Next Generation Technology for Global Software Development", #10-092313.

REFERENCES

1. Sara A. Bly, Steve R. Harrison, and Susan Irwin. 1993. Media Spaces: Bringing People Together in a Video, Audio, and Computing Environment. *Commun. ACM* 36, 1 (Jan. 1993), 28–46.

2. Alexander Boden, Frank Rosswog, Gunnar Stevens, and Volker Wulf. 2014. Articulation Spaces: Bridging the Gap Between Formal and Informal Coordination. In *Proceedings of the 17th ACM Conference on Computer Supported Cooperative Work & Social Computing (CSCW '14)*. ACM, New York, NY, USA, 1120–1130.

3. Michael Boyle, Christopher Edwards, and Saul Greenberg. 2000. The Effects of Filtered Video on Awareness and Privacy. In *Proceedings of the 2000 ACM Conference on Computer Supported Cooperative Work (CSCW '00)*. ACM, New York, NY, USA, 1–10.

4. Jed R. Brubaker, Gina Venolia, and John C. Tang. 2012. Focusing on Shared Experiences: Moving Beyond the

Camera in Video Communication. In *Proceedings of the Designing Interactive Systems Conference (DIS '12)*. ACM, New York, NY, USA, 96–105.

5. Gregorio Convertino, Dennis C. Neale, Laurian Hobby, John M. Carroll, and Mary Beth Rosson. 2004. A Laboratory Method for Studying Activity Awareness. In *Proceedings of the Third Nordic Conference on Human-computer Interaction (NordiCHI '04)*. ACM, New York, NY, USA, 313–322.
6. Fred D. Davis. 1989. Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS Q.* 13, 3 (1989), 319–340.
7. Paul Dourish and Sara Bly. 1992. Portholes: supporting awareness in a distributed work group. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '92)*. ACM, New York, NY, USA, 541–547.
8. Morten Esbensen and Pernille Bjørn. 2014. Routine and Standardization in Global Software Development. In *Proceedings of the 18th International Conference on Supporting Group Work (GROUP '14)*. ACM, New York, NY, USA, 12–23.
9. Robert S. Fish, Robert E. Kraut, and Barbara L. Chalfonte. 1990. The VideoWindow system in informal communications. ACM Press, 1–11.
10. Y. Ghanam, Xin Wang, and F. Maurer. 2008. Utilizing Digital Tabletops in Collocated Agile Planning Meetings. In *Agile, 2008. AGILE '08. Conference*. 51–62.
11. Stevenon Gossage, Judith M. Brown, and Robert Biddle. 2015. *Understanding Digital Cardwall Usage*. Technical report tr-15-01. Carleton University - School of Computer Science.
12. Sofiane Gueddana and Nicolas Roussel. 2006. Pêle-Mêle, a Video Communication System Supporting a Variable Degree of Engagement. In *Proceedings of the 2006 20th Anniversary Conference on Computer Supported Cooperative Work (CSCW '06)*. ACM, New York, NY, USA, 423–426.
13. Keita Higuchi, Yinpeng Chen, Philip A. Chou, Zhengyou Zhang, and Zicheng Liu. 2015. ImmerseBoard: Immersive Telepresence Experience Using a Digital Whiteboard. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*. ACM, New York, NY, USA, 2383–2392.
14. Jim Hollan and Scott Stornetta. 1992. Beyond Being There. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '92)*. ACM, New York, NY, USA, 119–125.
15. Steven Houben, Søren Nielsen, Morten Esbensen, and Jakob E. Bardram. 2013. NooSphere: An Activity-centric Infrastructure for Distributed Interaction. In *Proceedings of the 12th International Conference on Mobile and Ubiquitous Multimedia (MUM '13)*. ACM, New York, NY, USA, Article 13, 10 pages.
16. Hiroshi Ishii and Minoru Kobayashi. 1992. ClearBoard: A Seamless Medium for Shared Drawing and Conversation with Eye Contact. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '92)*. ACM, New York, NY, USA, 525–532.
17. M. Kuechler and A. Kunz. 2006. HoloPort - A Device for Simultaneous Video and Data Conferencing Featuring Gaze Awareness. In *Virtual Reality Conference, 2006*. 81–88.
18. M. Paasivaara, S. Durasiewicz, and C. Lassenius. 2008. Distributed Agile Development: Using Scrum in a Large Project. In *Global Software Engineering, 2008. ICGSE 2008. IEEE International Conference on*. 87 –95.
19. J. Paredes, C. Anslow, and F. Maurer. 2014. Information Visualization for Agile Software Development. In *Software Visualization (VISSOFT), 2014 Second IEEE Working Conference on*. 157–166.
20. Lene Pries-Heje and Jan Pries-Heje. 2011. Why Scrum Works: A Case Study from an Agile Distributed Project in Denmark and India. In *Agile Conference (AGILE), 2011*. 20 –28.
21. N. Roussel, H. Evans, and H. Hansen. 2004. Proximity as an interface for video communication. *MultiMedia, IEEE* 11, 3 (July 2004), 12–16.
22. Jessica Rubart. 2014. A Cooperative Multitouch Scrum Task Board for Synchronous Face-to-Face Collaboration. In *Proceedings of the Ninth ACM International Conference on Interactive Tabletops and Surfaces (ITS '14)*. ACM, New York, NY, USA, 387–392.
23. H. Sharp, H. Robinson, J. Segal, and D. Furniss. 2006. The role of story cards and the wall in XP teams: a distributed cognition perspective. In *Agile Conference, 2006*. 75–86.
24. Kar-Han Tan, I. Robinson, R. Samadani, Bowon Lee, D. Gelb, A. Vorbau, B. Culbertson, and J. Apostolopoulos. 2009. ConnectBoard: A remote collaboration system that supports gaze-aware interaction and sharing. In *Multimedia Signal Processing, 2009. MMSP '09. IEEE International Workshop on*. 1–6.
25. John C. Tang, Chen Zhao, Xiang Cao, and Kori Inkpen. 2011. Your Time Zone or Mine?: A Study of Globally Time Zone-shifted Collaboration. In *Proceedings of the ACM 2011 Conference on Computer Supported Cooperative Work (CSCW '11)*. ACM, New York, NY, USA, 235–244.
26. Jakob Zillner, Christoph Rhemann, Shahram Izadi, and Michael Haller. 2014. 3D-board: A Whole-body Remote Collaborative Whiteboard. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology (UIST '14)*. ACM, New York, NY, USA, 471–479.