Visualizing Progress Tracking for Software Teams on Large Collaborative Touch Displays

Brandon Scott-Hill, Craig Anslow, Jennifer Ferreira
Victoria University of Wellington, New Zealand
{craig,jennifer}@ecs.vuw.ac.nz

Martin Kropp, Magdalena Mateescu
University of Applied Sciences and Arts, Northwestern Switzerland
{martin.kropp,magdalena.mateescu}@fhnw.ch

Andreas Meier
Zurich University of Applied Sciences, Switzerland
andreas.meier@zhaw.ch

***Abstract*—Keeping track of the progress on software projects can be difficult and time consuming. Progress tracking requires developers to track progress by hand or digitally neither of which have good support for collaborative team processes. In this paper we present DashVis a tool to help support teams to track progress more effectively using large touch displays and visualization techniques. We conducted a study and found the visualizations to be very effective in supporting teams to gain a more accurate way of keeping track of progress. With large touch displays becoming ubiquitous in the work place and the demand for software teams to understand their progress more effectively there is a need for tools like DashVis.**

***Index Terms*—Collaborative Software Development, Visualization, Software Teams, Progress Tracking**

I. Introduction

Agile software development methods are used by 80% [10] of all software teams which brings benefits including team satisfaction [7], [21] and project success [16]. Digital tools have been created to better facilitate project management processes and practices including: GitLab [13], JIRA [4], Trello [5], and Monday [22]. Though software teams can still be seen using physical paper over digital tools [6], [14].

Keeping track of progress on software projects can be difficult and time consuming. Progress tracking visualizations have the benefits for communicating information around progress that otherwise may not be apparent. Some progress tracking visualizations exist [23] and teams that don't use visualizations tend to be inefficient [17]. Most progress tracking visualizations are designed for single user interaction but we need more effective tools to support team collaboration during meetings. Large touch displays offer affordances for teams to better support collaborative work flow [2]. Large touch displays have been used for software teams for planning and review meetings [8], [11], [12], [19], [20] and visualizing software artifacts [3]. However, no tools exist for visualizing progress tracking to support software development team meetings.

In this paper we introduce *DashVis* a tool to help make software progress tracking more effective for teams. DashVis uses large collaborative touch displays and progress tracking visualizations generated from sprint data in GitLab and Jira (see Figure 1). This paper addresses the research question, how effective are progress tracking visualizations on large collaborative touch displays for software teams?

978-1-7281-6901-9/20/$31.00 ©2020 IEEE

Fig. 1. DashVis on large touch display (65 inches) with visualizations: A) Burndown Charts, B) Cumulative Flow Diagram, C) Milestone Summary Chart, D) Parking Lot Diagram, and E) Niko Niko Calendar.

II. DashVis

DashVis is a new tool within aWall [19], [20] (a collaborative tool for software team meetings). DashVis contains a dashboard for different progress tracking visualizations displayed on large touch displays (65 and 85 inches). Developers can interact with the visualizations on the dashboard, add and remove visualizations, and arrange in different order. DashVis contains five visualization types (see Table I) where each has a different objective of what is being communicating to the team, with the intention to help identify problems that wouldn't have been able to be seen with raw data. Visualizations were selected based on how common they are [23] and feedback from earlier studies [19], [20]. DashVis obtains data from the GitLab [13] and JIRA [4] APIs, and is implemented in Angular, Interact, and D3 JavaScript libraries.

A. Burndown Chart

A burndown chart (Figure 1A) visualizes the number of tasks remaining over a period of time in a downwards fashion; it also includes a static linear line called the ideal line [1]. Burndown charts help communicate how teams are completing tasks by comparing to the ideal. If tasks are being completed

TABLE I
DASHVIS: VISUALIZATION TYPES.

Visualization	Type	Tracks Use	Agile Activity
Burndown	Progress	Workflow pace	Sprint Planning, Retrospectives
Cumulative Flow Diagram	Progress	Issue management	Sprint Planning, Retrospectives
Parking Lot Diagram	Progress	Issues left	Stand-ups
Niko Niko Calendar	Communicative	Team's feelings	Stand-ups
Milestone Summary Chart	Progress	Issue overload	Sprint Planning, Retrospectives

in a steady fashion close to the ideal line or are tasks being rushed to meet the deadline. Burndowns can also help teams understand if tasks are too small and being completed quickly or if tasks are too large and being completed too slowly. The implemented burndown chart uses a set of milestones as selectable data sets where the selected milestones due dates are used for the bottom axis and the milestones issues are used to work out when an issue was created and closed, allowing for data points of open issues on each day. Meaning that if developers don't use milestones, link issues to milestones or set due dates for milestones the visualization won't be able to create a burndown. The burndown has interactive features of a touch tooltip to display the amount of open issues of data points and animated transitions between selected milestones, sliding the axis in the direction of time and data points moving to new data points.

B. Cumulative Flow Diagram

The cumulative flow diagram (CFD) (Figure 1B) visualizes the number of tasks remaining in different categories over time in an upwards manner [25]. CFDs help compare differences between the amounts of work in various categories. Teams can use CFDs to help understand if they have too much work or if their backlog is filling up suggesting that they are feature creeping. The CFD was chosen due to being a unique form of progress tracking allowing for measurement of categories of work and their flow. The CFD uses the same set of milestones as the burndown where each selected milestone issue notes are checked when they were opened, closed, or had a label change; allowing for data nodes to render what states issues are on different days. Like the burndown chart, the CFD has the same requirements from the developer using GitLab in a certain way but also use GitLab's issue board by labelling issues and closing them. The interactive elements in the CFD include a touch tooltip for different amounts of tasks for each category in different data nodes and the axis have the same animated transitions as the burndown chart.

C. Milestone Summary Chart

The milestone summary chart (Figure 1C) visualizes all milestones, with the total amount of tasks for a milestone being the height of the bars, and the colour of bars being if a milestone is on time or not. The milestone summary communicates to teams if they are overloading milestones with too many issues and if it has impacted on meeting the deadline of a milestone or not. A milestone summary chart helps teams find a converging limit on the amount of issues that can be in a milestone and help to reduce over-promising for a milestone. The milestone summary uses project milestones and looks at each milestone's due date, total issues in a milestone, and when the last issue linked to the milestone is closed. The milestone summary chart interactive elements include a touch tooltip to display when the milestone was due, when the last issue was closed, if the due date has passed, and if there are still any open issues after the due date has passed.

D. Parking Lot Diagram

The parking lot diagram (Figure 1D) visualizes the number of tasks leftover over total tasks, displays the amount of days left, and uses colour to display if tasks are done (green), close to the deadline (red), in progress (blue), and not started (white) [15]. The parking lot diagram helps teams understand how much work is left and how much time they have to complete it, even if they want to ignore how close they are to a deadline. The parking lot diagram was chosen because of it's simplicity of tracking progress. The parking lot diagram gets the sets of milestones where the selected milestone due dates are used to work out how many days are left and gets stats of the selected milestone around how many issues are left open. Like the other visualizations it requires using the deadline system in GitLab and linking issues to a milestone. This visualization doesn't include any interactive features due to its simplicity.

E. Niko Niko Calendar

The Niko Niko calendar (Figure 1E) or smiley calendar communicates team status instead of being a progress tracking visualization [24]. The Niko Niko calendar visualizes how different team members are feeling for different days of the week with the use of smileys. It can help communicate if team morale is low which could be related to current work going in the project or helps identify if a team member is having issues. Teams can use Niko Niko calendar responses to identify team members that are having problems and help that team member whether it be emotional and or technical. The Niko Niko calendar was chosen because it illustrates team mood which is an important factor for successful teams. The Niko Niko calendar uses the set of milestones to select a milestone where the start and due dates of the milestone makes a data range and obtains a list of members of the GitLab project. That data is then filtered based on user input as not every member of the GitLab project is part of the software team, nor do they work every weekday. Like the other visualizations the Niko Niko calendar relies on using due dates on milestones. The calendar interactive elements are tapping on a spot to cycle through the different smileys.

F. User Interface

The dashboard is an empty area for all the visualizations and is the main screen of DashVis. The dashboard includes buttons

on the top right to control the modes (e.g. view and edit) of the dashboard and add more visualizations. Visualizations can be dragged around the dashboard, placed anywhere on the display, and can be snapped together in a grid form. Each visualization uses a wrapper which handles all user interaction on the dashboard such as selection of data by dropdown and dragging of the visualizations. In the view mode users cannot move visualizations around nor delete them. In the edit mode users can move visualizations around, remove them with the minus button, control settings, and add visualizations.

III. User Study

To evaluate the effectiveness of DashVis on large touch displays we conducted a qualitative user study to understand how the tool could be used for software development. We recruited two student teams during a two week sprint to help keep track of progress which meant they had to use the tool for the entire duration. The students were from our 300 level group project software course and each team had five members. The student teams were a convenience sample as they both had an existing project and they were quite actively engaged in the process of managing a project given the course they were undertaking. The two teams worked with DashVis for a single sprint for two weeks. Due to the nature of the course students worked on their project in two weekly four-hour lab sessions per week meaning each team participated in the study for a total of 16 hours (Table II). Each participant received a $10 voucher. The study took place in an open plan room with computers and the large touch display as the central focus.

For the procedure each participant was given an information sheet to read, a consent form to complete, and could ask questions during that process. Teams then used DashVis for the sprint. At the end of the study each participant completed a questionnaire. The idea was to put teams in an environment where they could track project progress using a large touch display during a sprint. The point was to not control how teams used DashVis in the study but let them decide when to use the tool. Teams were aware they were participating in a user study which influenced them to use DashVis. The questionnaire had three different types of questions to understand the perceived effectiveness of the techniques (See Appendix). Likert Scales (range 1 to 5, where 1 being very ineffective to 5 being very effective) [18], Software Usability Scale (SUS) [9], and open ended questions were use to collect data.

IV. Results

We now present the results from the questionnaire to address the research question. A total of 10 responses were collected from 10 participants from the two teams.

TABLE II
User study participants & length

Team	Participants	Sessions	Time
A	5	4	16 Hours
B	5	4	16 Hours
Total	10	16	32 Hours

(a) Value Results. Q1 - blue, Q3 - orange, Q12 - grey.

(b) Validity Results. Q5 - orange, Q6 - blue.

Fig. 2. Questionnaire Results.

Questions 1-4, and 12 helped to answer is there is any value to using DashVis on projects. In particular what value did teams feel that they got out of using DasVis, and if they deemed DashVis to be valuable for use in other projects (Figure 2(a)). These questions also helped to understand if DashVis improves the ability to manage their projects. This also works the other way around, if DashVis didn't help with their ability to manage how could DashVis have benefited their project. The results indicated that participants gained an average value to their ability to manage, and their project gained an average value from using progress tracking visualizations. Participants on average deemed that progress tracking visualizations was useful enough that they would likely want to use them in other projects. The results suggest that participants found that the progress tracking visualizations to be of value on a large touch display to the point that they are willing to keep using them.

Questions 5 and 6 helped to confirm that participants understand the visualizations (Figure 2(b)). If participants don't actually use DashVis or understand the visualizations, it would be difficult to comment if DashVis has any value or how effective the visualizations were. The results showed that participants perceived they had a competent understanding of the visualizations overall and frequently used the visualizations.

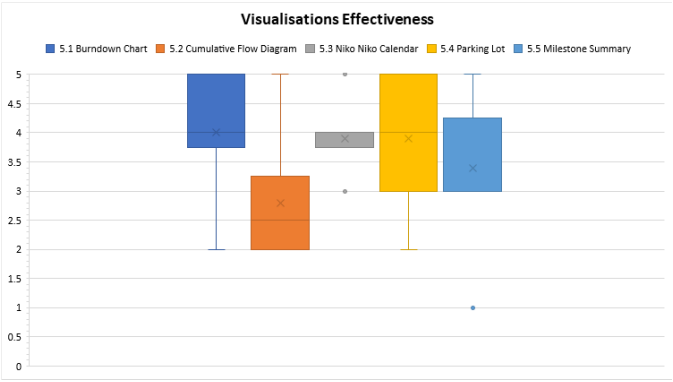

Fig. 3. Questionnaire Visualization Effectiveness Results. Burndown Chart - dark blue, Cumulative Flow Diagram - orange, Niko Niko Calendar - grey, Parking Lot Diagram - yellow, Milestone Summary - light blue.

Questions 7-11 helped to answer the effectiveness of each visualization on large touch displays (Figure 3). The results showed that participants on average perceived that the Burndown Chart, Parking Lot, and Niko Niko calendar were the most effective and the Cumulative Flow Diagram was the least

ineffective. The Burndown Chart, Parking Lot, and Niko Niko Calendar were effective as the diagrams are easily to explain what is happening at first glance and had intuitive features for interaction. The Milestone Summary Chart had an average effectiveness which could have been attributed to the limited number of sprints in the study so hard to make comparisons over a short period of time. The CFD was ranked the lowest because participants didn't understand how to read the diagram and they needed further training on that technique.

The System Usability Scale [9] was used to measure the usability of DashVis (Figure 4). For question 14 participants answered the 10 SUS questions. DashVis achieved a system usability of 68% which is considered an average level of system usability. The highest score was 85% and lowest 45% which suggests that the usability experience was very good for some participants and not very good for others. Of note question eight "I found the system very cumbersome to use", scored below 2 which can be attributed to hardware issues caused by the accuracy of the touch detection on the display.

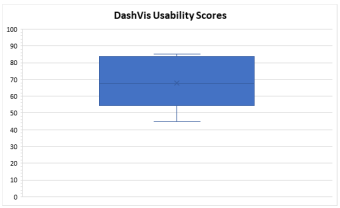

Fig. 4. Questionnaire System Usability Scale Results

V. Discussion

The user study went for a total of 32 hours and collected ten completed questionnaires. In that time participants used DashVis as they worked on their sprint and provided feedback. The results allowed for checking response validity, examine the system usability of DashVis, understand the value of DashVis, and find which visualizations were most effective.

It was found that participants perceived that on average the Burndown Chart, Niko Niko Calendar, and the Parking Lot to be effective, with most participants thinking that the Burndown Chart to be the most effective. While at the same time participants perceived the Cumulative Flow Diagram to be the worst with their justifications being, they didn't know how to interpret the diagram. The issue here is participants needed more training on how to understand the CFD which may have made it more effective. To comprehensively find find which visualizations are most effective they need to have similar usability and to be used over a much longer period of time. There is value knowing which visualizations are most effective for teams to help inform future designs of collaborative team based progress tracking software.

The user study questionnaire also focused on how much value participants got out of DashVis and how much value did their project get. From the results, participants perceived that they got some personal value and that value translated to their projects. What was important to check was if they did get value out of DashVis, were they willing to use it for other projects. The results indicated that participants who were novices to software development and project management got some value out of DashVis and suggested that DashVis does have potential if expanded upon the tool that could give real-world value to developers to manage their projects more effectively, and possibly make their project more successful.

The participants recruited for the user study were a convenience sample of novices with both limited in numbers and experience, as there was only 10 participants from the two teams. For stronger results it would require more teams, people, and used for a longer period. The participants are also very similar and have similar experiences that does not accurately reflect industry. The nature of the recruited teams only worked for 8 hours a week per person, which doesn't reflect the amount of time typical industry teams work on a weekly basis. This limitation of teams weekly work length means that the sprints in hours are extremely short being only 16 hours sprints over 2 weeks, while realistic agile teams sprints would be more around 40 hours per week. Evaluating with professional software developers was out of scope for this project.

VI. Conclusions

Agile software development teams often use tools for tracking progress like Jira and GitLab but they are limited in visualization capabilities, often focus on a single user display & interaction, and lack support for collaborative team practices. In this paper we introduced *DashVis* a tool to help make software progress tracking more accessible for teams. DashVis is part of a much larger project called aWall [19], [20]. DashVis uses large collaborative touch displays and novel progress tracking visualizations integrated into software team projects. DashVis is a tool that contains five different types of visualizations: Burndown Chart, Cumulative Flow Diagram, Niko Niko Calendar, Parking Lot Diagram, and Milestone Summary. Multiple visualizations can be displayed at once showing data from the same or different sprints. These visualizations use data from GitLab and Jira and have been integrated with the aWall project. To evalaute the effectiveness of DashVis we conducted a user study with two student teams 10 participants each in total which went for a length of 32 hours. The results found that the Burndown Chart, Niko Niko Calendar, and the Parking Lot Diagram were the most effective techniques and participants preferred the Burndown Chart due to the rich nature of information it provides. The results also indicated that there was value in tracking progress about project teams on large collaborative touch displays. Our goal is for developers to use tools such as aWall and DashVis in the near future as large touch screens become ubiquitous in the work place. Future work would be refining some of the developed visualizations, implementing some different visualization techniques based on feedback in this user study, and conducting a longitudinal study with professional software developers.

ACKNOWLEDGEMENTS

We thank the student teams for participating in this research project and for experimenting with our software prototypes.

REFERENCES

[1] Agile-Alliance. What is a burndown, 2020. Retrieved from https://www.agilealliance.org/glossary/burndown-chart/.

[2] Craig Anslow, Pedro Campos, and Joaquim Jorge, editors. *Collaboration Meets Interactive Spaces*. Springer, 2016.

[3] Craig Anslow, Stuart Marshall, James Noble, and Robert Biddle. Sourcevis: Collaborative software visualization for co-located environments. In *Proceedings of International Conference on Software Visualization (VISSOFT)*, pages 1–10. IEEE, 2013.

[4] Atlassian. Jira, 2020. Retrieved from https://www.atlassian.com/software/jira.

[5] Atlassian. Trello, 2020. Retrieved from https://trello.com/.

[6] Gayane Azizyan, Miganoush Magarian, and Mira Kajko-Matsson. Survey of agile tool usage and needs. In *Proceedings of Agile*, pages 29–38, 2011.

[7] Robert Biddle, Andreas Meier, Martin Kropp, and Craig Anslow. Myagile: sociological and cultural effects of agile on teams and their members. In *Proceedings of International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE)*, pages 73–76, 2018.

[8] Andrew Bragdon, Rob DeLine, Ken Hinckley, and Meredith Morris. Code space: Touch + air gesture hybrid interactions for supporting developer meetings. In *Proceedings of the International Conference on Interactive Tabletops and Surfaces (ITS)*, pages 212–221. ACM, 2011.

[9] John Brooke. System usability scale, 1986. Retrieved from https://www.usability.gov/how-to-and-tools/methods/system-usability-scale.html.

[10] CollabNet. 14th Annual State Of Agile Report, 05 2020. Retrieved from https://explore.digital.ai/state-of-agile/14th-annual-state-of-agile-report.

[11] Morten Esbensen, Paolo Tell, Jacob Cholewa, Mathias Pedersen, and Jakob Bardram. The dboard: A digital scrum board for distributed software development. In *Proceedings of the International Conference on Interactive Tabletops and Surfaces (ITS)*, pages 161–170. ACM, 2015.

[12] Yaser Ghanam, Xin Wang, and Frank Maurer. Utilizing digital tabletops in collocated agile planning meetings. In *Proceedings of Agile*, pages 51–62. IEEE, 2008.

[13] GitLab. Gitlab, 2020. Retrieved from https://gitlab.com.

[14] Stevenson Gossage, Judith Brown, and Robert Biddle. Understanding digital cardwall usage. In *Proceedings of Agile*, pages 21–30, 2015.

[15] Mike Griffiths. Summarizing progress with parking lot diagrams, 2007. Retrieved from https://leadinganswers.typepad.com/leading_answers/2007/02/summarizing_pro.html.

[16] Standish Group. The chaos report, 2019. Retrieved from https://www.standishgroup.com/.

[17] N. Hajratwala. Task board evolution. In *In Proceedings of the Agile Conference*, pages 111–116, 2012.

[18] Susan Jamieson. Likert scale, 2017. Retrieved from https://www.britannica.com/topic/Likert-Scale.

[19] Martin Kropp, Craig Anslow, Magdalena Mateescu, Roger Burkhard, Dario Vischi, and Carmen Zahn. Enhancing agile team collaboration through the use of large digital multi-touch cardwalls. In *Proceedings of the International Conference on Agile Software Development (XP)*, pages 119–134, 2017.

[20] Martin Kropp, Judith Brown, Craig Anslow, Stevenson Gossage, Magdalena Mateescu, and Robert Biddle. *Interactive Digital Cardwalls for Agile Software Development*, pages 287–318. Springer, 2016.

[21] Martin Kropp, Andreas Meier, Craig Anslow, and Robert Biddle. Satisfaction, practices, and influences in agile software development. In *Proceedings of the International Conference on Evaluation and Assessment in Software Engineering (EASE)*, pages 112–121. ACM, 2018.

[22] Monday. Monday, 2020. Retrieved from https://monday.com/.

[23] Julia Paredes, Craig Anslow, and Frank Maurer. Information visualization for agile software development. In *Proceedings of International Conference on Software Visualization (VISSOFT)*, pages 157–166, 2014.

[24] Akinori Sakata. Niko niko calendar, 2006. Retrieved from https://sites.google.com/view/niko-niko-calendar/home/en.

[25] Yodiz. Cumulative flow diagram (cfd), 2020. Retrieved from https://yodiz.com/help/cumulative-flow-diagram-cfd/.

APPENDIX

1) How valuable was tracking project progress on the digital touch display to your ability to manage your project? (1 - No value, ... 5 - Extreme value)
2) Why do you feel this way?
3) How valuable was tracking project progress on the digital touch display to your project? (1 - No value, ... 5 - Extreme value)
4) Why do you feel this way?
5) How competent do you feel understanding these visualizations? (1 - Not competent, ... 5 - Highly competent)
6) How often did you use the visualizations? (1 - Hardly ever, ... 5 - Almost always)
7) How effective was the Burndown Chart visualization for tracking progress? (1 - Very ineffective, ... 5 Very effective)
8) How effective was the Cumulative Flow Diagram visualization? (1 - Very ineffective, ... 5 Very effective)
9) How effective was the Niko Niko Calendar visualization? (1 - Very ineffective, ... 5 Very effective)
10) How effective was the Parking Lot visualization? (1 - Very ineffective, ... 5 Very effective)
11) How effective was the Milestone Summary visualization? (1 - Very ineffective, ... 5 Very effective)
12) How likely will would you want to use progress tracking visualizations in your next project? (1 - Very unlikely, ... 5 - Very likely)
13) Which is better for tracking project progress a large touch display or normal non-touch display, and why?
14) Please answer the following questions from the SUS [9].
15) Do you think there is any missing visualizations in the progress tracking dashboard?
16) Are there any features that you think that are missing or could be improved?
17) Any additional comments?