VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wānanga o te Ūpoko o te Ika a Māui

School of Engineering and Computer Science
Te Kura Mātai Pūkaha, Pūrorohiko

PO Box 600
Wellington
New Zealand

Tel: +64 4 463 5341
Fax: +64 4 463 5045
Internet: office@ecs.vuw.ac.nz

aWall: Collaboration Support for Agile Retrospectives

Simon Glew

Supervisor: Dr. Craig Anslow

Submitted in partial fulfilment of the requirements for
Bachelor of Engineering with Honours.

Abstract

This is a problem as it means either product teams have to either be co-located or have to be all parts of agile meetings using a third party meeting tool such as Skype. Integrating an agile retrospective method within aWall will allow users to do all agile meetings within one online agile tool.

Chapter 1

Introduction

1.1 The Problem

The problem that this project is to solve is attempting to support agile retrospectives meetings through touch walls and screens, this is going to be done by creating a prototype application that will be used to support them. This project is part of the aWall software project run by Dr. Craig Anslow and Professor Martin Kropp of FHNW in Switzerland. aWall is Agile Retrospectives are meetings held within teams once complementing an increment of work. They allow teams to inspect and adapt the different areas that are required within themselves and allow for change and learning within the entire team [1].

The outcome of this project, is to build a software web system to help facilitate agile retrospective meetings using a large touch screen as an output for the software. Once this software has been built, an evaluation with users will occur to evaluate and improve on the software project. The current aWall project can be found here: `https://www.youtube.com/watch?v=fzCnjnpRiTI`

Figure 1.1: Project aWall - digital agile cardwall being used [6]

1.2 Overview of Research Project

This project will be building off this current prototype but within its own project scope and no integration between the two codebases.

This software prototype is split into three different components:

- **Participant System:** The participant system is used for the participants to interact within the system, it is an application that will allow the participant to vote and make notes about the retrospective and directly interact with what is happening on the screen and within the retrospective. All the interactions from this system get seny and stored within the database housed within the server. This system will be accessed from a secondary device by the user e.g their phone. This system sets up a connection to the server through a socket, allowing for the required transfer of data.

- **Screen System:** The screen system is an application that the moderator of the retrospective interacts with, it allows the moderator to display and manipulate the data passed from the server that the participants have given. This system will be used on the touch screen with a connection to the server through a socket.

- **Server:** The server houses all the data storage and manipulation for the overall software system. The server also houses the socket system allowing for real-time updates between the participant and screen systems. The server allows the storage of data for futher use in later iterations of the projects lifecycle, e.g. a later agile retrospective.

Within this prototype there will be different versions of agile retrospectives that the different members of the agile team can choose. The different retrospectives that were chosen through doing a background review on the different retrospectives methods that where found. The ones that were chosen either linked up with the other two sections of a retrospective nicely or were found within multiple different sources [1, 5]. The retrospective methods that were chosen are:

- The 3W's/Mad, Sad and Glad
- Timeline
- Brainstorm/Filtering
- Short Subjects

Each of these retrospective methods will be able to be selected from the screen system within the software application when creating a retrospective session. To accompany each of these retrospective methods, there will also be short sections before and after the main method, it allows for the setting of the scene within the retrospective and to wrap up the retrospective also.

The methods that I have chosen for this are: [1]

- **Setting the scene:** Check-in
- **Wrapping-up:** +/- or Delta

Using each of these methods, I will be able to evaluate the best retrospective method through conducting user testing using all of the methods above.

Chapter 2

Background Review and Related Work

Retrospectives are meetings held within agile teams, involving all members of the team and is held at the end and just after an iteration of work [1, 3]. The retrospective is used for not only celebrating the success of work from the last retrospective, but also the failures and lessons that can be learnt from them [5]. The retrospective is normally facilitated by a third party member who does not have a personal stake in the in the content or outcome of the meeting, therefore being able to remain neutral during the meeting [5, 7]. Retrospectives are used to reflect on the work done since the last retrospective and the problems that the team faced within this work. [1]. aWall is a online tool that allows for collaboration within a team when it comes to agile practices [6], including the aqile practice of retrospectives. With the tool being online, it allows teams to seperate members while still all members can include thoughts and feelings to the agile practices within the team.

NEEDS MORE WORK - WILL DO LATER THIS WEEK, WITH SPLIT INTO SECTIONS

2.1 Agile and Agile Retrospectives

das

2.2 aWall

dsad

Chapter 3

Work Done

The work currently accomplished within the project can be seen in the sections below:

3.1 Technical Work

3.1.1 Design

The current design of the prototype can been seen below:

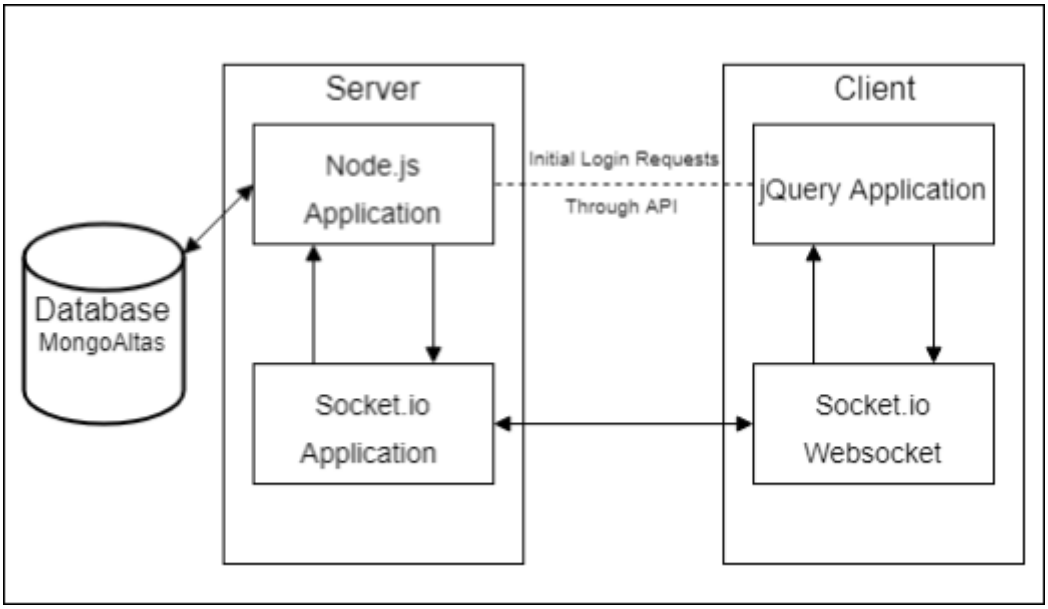

Figure 3.1: Current Design Framework of aWall Retrospective Prototype

As we can see from the picture in Figure 3.1, the participate and screen systems have direct links to server through both the sockets and through requests. The need for the socket connections to go through the server is useful for data store and manipulation before being send out, allowing for more calculation and processing strain on the server rather than the users device. The need for both the socket communication and requests between the server and other systems is due to the need for joining and create retrospective boards within the system. The sockets are stored within a map on the server, with needing a name that is given on joining a session to properly store the data. Therefore due to this, the need for requests is required to send data when joining a session.

The connection between the database, sockets and the server is setup during the startup of the server and remains open and available during the entire time the server is alive.

3.1.2 Design Choices

The technologies chosen for this project are: a jQuery frontend application, with a node.js/socket server using mongoDB as a datastore within the application.

- **jQuery:** [4] This was chosen as the front-end framework for the application on the advisement of my project supervisor. This is due to the current technologies that are housed within the aWall application. jQuery was one of the technologies that was picked that is currently housed within the overall aWall prototype, allowing for simple integration between this prototype and the current aWall prototype.

- **Node.js:** [2] Node.js was chosen as the backend framework due to its ability to quickly setup and integrating into other parts of the prototype through libraries such as *Socket.IO* [12] for Sockets and *Mongoose* [10] for MongoDB. The aWall prototype does not use Node.js as a technology, its backend framework is *Python* [11], this wasn't chosen due to the complexity of integration with the other technologies.

- **Sockets:** [12] Sockets were needed for this project due to the need of real-time communication within the prototype between the participant and screen systems. This was chosen due to it being one of the technologies within the current aWall prototype, therefore making it easier to integrate the two prototypes together.

- **MongoDB:** [9] MongoDB was chosen as the datastore for the prototype, this is not the datastore held within the current aWall project, as that datastore will not allow me to store the correct data needed for the retrospective section. *MongoDB*, more specifically *MongoAtlas* [8] was chosen due to it's simplicity and quickness to setup and get something running. It also allows for easy integration within the backend framework with the library *Mongoose* [10].

With using both jQuery and Socket.IO as the major libraries within the frontend development of the application it has will make it possible to very easily replace the backend that has been developed within this prototype and integrate this frontend into the current prototype of aWall that uses both of these technologies with next to no changes except routing points within the jQuery application.

Through the planning and start of the implementation, there has been solutions that have been looked over and removed from within the project due to not fitting within the scope required of them in the project. These are:

- **Datastore Choices:** The original aWall system used JIRA as a datastore system, connecting through their API to get the required data. JIRA was used as it was easy to integrate and get the required data for the other stages of the agile framework such as sprint planning and reviews. JIRA does not support a datastore for retrospectives, therefore it was removed as a possible solution for the datastore of the project. Due to not being able to use the datastore that was in the previous system, I have to use some form of database instead. The choice of using MongoDB was found due to its easy ability to store the required variable data that will be given from users, that I wouldn't be able to get with using a SQL database. With the use of MongoDB, I found that the use of MongoAtlas was good for the solution as it then meant I didn't need to host it locally due to that being done with Atlas.

3.1.3 Current Implementation Progress

The current implementation progress within this system is as follows:

- **Server:** The server has been fully setup with its socket counterpart and its connection to the datastore MongoDB. It currently has built within it a framework that allows for easy ability to add and remove extra retrospectives within itself, with 4 of the 6 retrospective types having this completed. The server is also being current hosted within the ECS system at Victoria University and can be found at: `http://barretts.ecs.vuw.ac.nz:52724/`
- **Touch:** Within the touch system, the header component with its required scope file working and also one of the retrospective methods. This header file is constant throughout the entire retrospective for the screen, it displays the most important data throughout the entire retrospective such as time elapsed, members and current stage. The work for the first retrospective type, Check-in has also been completed, with connection to the server to grab the required data for it.

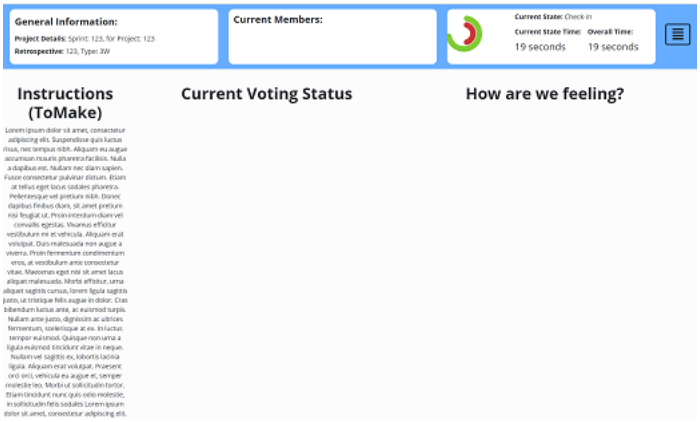

Figure 3.2: Current Progress of Screen System within aWall Retrospective Prototype

Figure 3.2 shows the current progress that has been made on the touch system as of the 3rd of June. Currently implemented within it is the header, showing all the required information such as the retrospective details, the time elapsed and the current members within the retrospective. The view that is currently shown is the only one that has been implemented, this is the check-in view for the moderator showing the three different sections that will be viewed.

Figure 3.3: Current Progress of Header within Screen System within aWall Retrospective Prototype

Figure 3.3 shows the current progress that has been made on the header as of the 3rd of June. It is split into the three sections, each containing the required information being displayed for the team of the agile retrospective. It contains the Project details such as the sprint number and project name of the retrospective, the current members connected to the board and finally the time elapsed between the current state and the overall time of the retrospective.

- **Participant:** The participant view, the only progress that has been made has been within it has been the voting system for the first retrospective and the connection to server to write the data to the datastore.

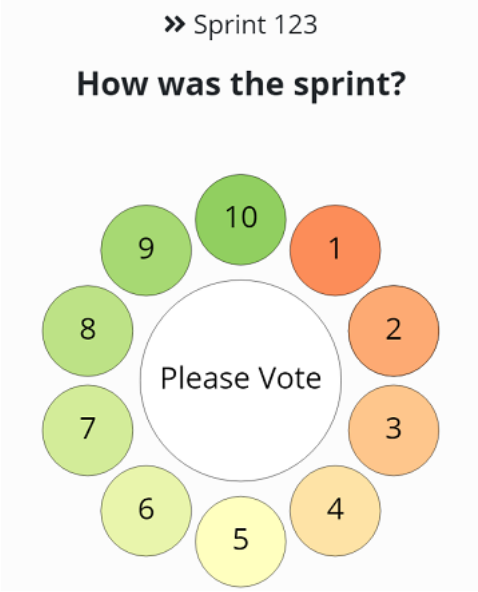

Figure 3.4: Current Progress of Participant System within aWall Retrospective Prototype

Figure 3.4 shows the current progress that has been made on within the participant system with aWall. This system currently is being built with a phone in mind to interact with, but implementation will include a desktop view later on. This is the check-in view, allowing the user to vote within the retrospective.

3.2 Evaluation of Solution

The solution is going to be evaluated through User Studies. A ethics proposal has been submitted to the Human Ethics Committee within Victoria and is currently going through the process of being evaluated and approved, the ethics application number is *#0000026187*. Within this user study I am looking to test on 4 -5 teams, from teams within the ENGR301/302 courses.
The process for the user study will consist of the following steps:

- Users will volunteer and be placed into groups or 'teams' to do the retrospectives in.
- The teams will firstly participate within a 'control' retrospective, where they will do part of a retrospective the old way, using a cardwall.
- The teams will then participate in a retrospective using the aWall tool using one of the 4 main retrospective methods
- Finally the members within each of the teams will fill out a questionnaire about the experience of using the aWall prototype compared to the cardwall retrospective.

Using the data gathered from the questionnaires and from the observations within the user study, I will be able to not only evaluate the effectiveness of using a touch wall over a card wall, but also what retrospective method has the greatest engagement within different software teams.

Chapter 4

Future Work

4.1 Project Components left to complete

1. Finish development of all three components with the different retrospective types
   By Mid July
2. User study preparation
   By 1st August
3. Conduct User Study
   By 31st August
4. Analysis of User Study
   By 20th September
5. Clean up Preferred System with Software and integration within aWall prototype
   By 30th September
6. Write Final Report
   By 26th October

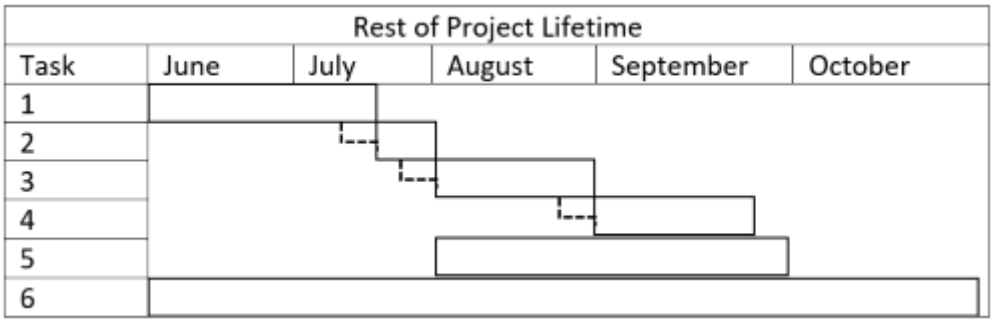

Figure 4.1: Current timeline planned for the project

Bibliography

[1] DERBY, E., AND LARSEN, D. *Agile Retrospectives: Making Good Teams Great*. Pragmatic Bookshelf, Raleigh, North Carolina, 2006.

[2] FOUNDATION, N. Node.js, 2018.

[3] GONCALVES, L., AND LINDERS, B. *Getting value out of Agile Retrospectives: A Toolbox of Retrospective Exercises*. Leanpub Publishing, Victoria, British Columbia, 2013.

[4] JS.FOUNDATION, J. jquery, 2018.

[5] KEITH, N. *Project retrospectives: a handbook for team reviews*. Dorset House Publishing, New York, New York, 2001.

[6] KROPP, M., ANSLOW, C., MATEESCU, M., BURKHARD, R., VISCHI, D., AND ZAHN, C. Enhancing agile team collaboration through the use of large digital multi-touch cardwalls. *LNBIP*, 283 (2017), 119–134.

[7] LINDERS, B. 7 retrospective facilitation good practices, 2016.

[8] MONGODB. Mongo atlas, 2018.

[9] MONGODB. Mongodb, 2018.

[10] MONGOOSE. mongoose, 2018.

[11] PYTHON.ORG. Python, 2018.

[12] SOCKET.IO. socket.io, 2018.