SEM5640 Group Project
Individual Report

Go!Aber

Submitted in partial fulfillment of
the requirements for the award of the degree of

MEng
in
Software Engineering

Submitted by

110005643 Craig Heptinstall

Department of Computer Science

January 2016

Contents

Chapter 1

Introduction

This report looks into my personal views of the group project completed for the Go!Aber application. I will be evaluating the group performance, as well as my own personal performance in the following two chapters. I will provide critique and possible improvements for various times in the project, outlining what I felt when well, and what could have gone better. Throughout this evaluation, I will be using my personal blog that I wrote each week. The blog entries can be found in appendix A, or on my blog site [1].

1.1 Personal evaluation

To start by looking at my own contribution to the project, this began during the initial reading of the requirements specification. Alongside the other group members, I carefully read through and provided description about each specification, noting down any possible questions or issues. Following the minutes recorded (this was done by myself throughout the project), questions recorded were then put to the client. Again, I was responsible for creating the minutes from that meeting, and were placed in the project folder for access by any other members, client or project manager.

Staying on the subject of my role during meetings, in each sprint review with the project manager, I ensured that both the meeting room was free and also that major statistics such as burndown, sprint tasks completed, and hours worked were explained. This was an important task for me, as I was selected as scrum master. I was the member to suggest using scrum as the principle development methodology, following from my personal experiences during my industrial year.

[1] `http://craighep.blogspot.co.uk`

I felt that both scrum, and my role as scrum master were both handled well, and that I ensured at the beginning and end of each sprint that the team as a whole were on track to complete the weeks tasks. Perhaps one improvement that could have bettered my performance as scrum master could have been to explain the roles and rules of the methodology better to other members after the workflow was selected. This could have improved the use of TFS (Microsoft visual studio online), where instead some members were slightly late on creating new tasks, and marking others completed when required.

Furthering the scrum responsibilities I held, I am happy with the way that I managed meetings for sprint planning. Where meetings had time restraints, I ensured that the team did not discuss certain issues for too long (unless necessary). For instance, in initial discussions of the class diagram, I suggested that because we were still only at the stage of designing the database, that the team should not get too far ahead, and take more time concentrating on current work.

Finally for my scrum master role, I worked closely with all other team members to ensure that the most suitable tasks were placed with a higher priority, therefore ensuring these tasks were completed first. For instance, creating a stable database was the first task that I agreed had to be completed before any other task could be started.

Another couple of tasks that I knew had to be completed early on in the project was the development of the Fitbit and Jawbone connections. Because I had some experience with API connections and SOAP/ Rest connectivity, I put myself forward for handling these tasks. Starting by browsing a few examples on the internet and some that were suggested in the practical lectures of this module, I was able to create external projects capable of connecting to the Fitbit servers from .NET. This was completed within the second week of the project, and the first sprint week.

Carrying this over, I was then able to create the same connection within the application .NET project. Although I managed to get the .NET connection working, the Java half was not started at the end of the first sprint. Mainly due to inexperience with the Fitbit API working with Java, I was unable to start this until sprint two, and this then pushed back my ability to start the Jawbone work. Therefore, I asked help from a fellow member to complete the Fitbit connection, and put to use my experience of pair programming.

Although I enjoyed this way of development, and progress was made faster than by myself, we were unable to fix the issue with Fitbit on Java, and

therefore decided as a group (though advised by me) to move onto another task until more free time could be made to complete the Java connection. Though I felt that I could have prepared better for implementing the Java side of this task, I knew that there were still integral parts of the system to be completed.

In the following sprint I handled the task to control system permissions (revoking and checking connections) with Fitbit and Java. This task was completed on time in this case, and I also managed to help a fellow member with Jawbone .NET connectivity. The following sprints then consisted of me cleaning up the connectivity by refactoring and commenting, whilst also attempting to resolve the issue with Fitbit and Java.

Overall, I felt that the contribution towards the completed application was what was required, although I did not complete all my tasks. If I were to perform the same tasks again, then I would ensure my knowledge of API connectivity in Java was better. Also, if I knew the available libraries that I know now to help with such tasks, then the productivity of my task completion would be much faster. Even with my knowledge of using APIs with both languages from previous work experiences, I found that issues I may have had in the past are prevented. This project was a matter of refreshing my knowledge of writing good web applications.

Following on from this, during the initial stages of the project it was agreed that each member should apply the same coding standards to ensure consistency and readability. I felt that I stuck to the coding standards for each language well, and refactored after I completed tasks where necessary. I do feel however that I could have placed more comments in some parts of the work I completed, to allow other members to understand my code better, and also for the project manager or client if they wished to read into any of the code. I did ensure that all important methods in the code did have a description in terms of Javadoc or the .NET documentation where needed.

As git was decided as our version control management system, I was happy with this because of my previous history using it. By creating task specific branches, I was able to work easily without affecting others work, although when it came to pulling the changes through to the common branch, I did require help from others at times. I felt that I communicated well, and ensured that any changes I pulled through would not affect others work, and that if I saw any big changes to the common branch, I would then question the member who had made those changes.

Throughout the project, I also provided assistance to others where they needed advice on implementing certain features, such as the Jawbone connection, or help with the scheduler implementation concerning how often to update certain parts of the system. Although I feel I did not perform enough code reviews of others while in pull requests, I made up for this in the latter half of the project, and in most sprints reviewed and gave feedback on all tasks completed by others.

Once the majority of tasks were completed, I then volunteered to perform cucumber testing on both projects, initially starting with the .NET application. Because I had lots of experience cucumber testing in the past, I let the other members of the team know that I would be best suited to this task. After trying out a few libraries, I found one that would work for both the .NET and the Java application, cutting down work time. Rather than creating two sets of tests, I was therefore able to use the same set for both applications, and write extension user interaction tests.

I was happy with the amount of tests and detail I performed, though as always with cucumber testing, more could have been added. If I had known about which library to use, and that a library could be used for multiple purposes at the start of the project, I could have created many more tests, and added more comprehensive boundary and extreme testing. I ensured that testing was completed with time to spare, to allow myself to complete documentation, and allow other team members to write about testing correctly in their sections of the report where necessary.

Due to the issues with Fitbit and other smaller setup issues at the beginning of the project, alike the other group members I did find myself behind schedule overall in the project, resulting in the documentation to be started quite late on. We initially planned for the project implementation to be completed at least a week before hand in, but instead I found myself writing the documentation only 3 or 4 days before hand in.

Even with this issue though, I was pleased to complete this a day early of the deadline, with time to create a skeleton project document for the other group members to add their own documentation to. Because I created the document, I also volunteered to clean up the document when all sections were added. My section (introduction and requirements) was proofread, while I proofread others. I selected to read through the testing documentation especially, as I had done quite a lot of testing (primarily cucumber).

At the end of documentation creation, I then did a general tidy of the docu-

ment, and ensured that the group as a whole were happy with its final state. Once complete, I then handed this to the member who was responsible for submitting the project. Although I ensured I did not take too long creating the document layout, I made sure it was made to a very good standard.

In addition to the document, I also created a lot of the diagrams for the report and for the project manager to look at during the design stages. I chose to do this because during one sprint, I found myself completing my tasks for that week, therefore having time to create any UML diagrams required.

As I have had plenty of experience in the past creating UML, I feel that I have created a comprehensive set of diagrams explaining most the major components and usability features of the application. I also went through each of my diagrams with the other members on completion to make sure that the others knew what each diagram was for, so that in the documentation stage of the project the diagrams could be used and described.

In terms of keeping time and maintaining the schedule we set out, although this started out rather negatively, I have been able to improve vastly over the second half of the project. I noticed that as I had completed tasks, I then knew what to expect in coming ones, therefore making it easier to assess the required workload for future tasks.

The blog I created to accompany this report was done on a weekly basis, and helped me greatly during the writing of this report, and also in providing useful information to both the project manager and the other group members in terms of my work efforts.

1.2 Group evaluation

In addition to reviewing my own performance and contributions to the group project, it is important to review and understand what and why specific parts of the project as a whole went well and did not go so well. To do this, I will evaluate each stage of the project and talk about how well I thought the team completed it.

Upon initially receiving the project, we set up a means of communication through Facebook and by exchanging phone numbers. I thought this was good, and made sure that all members were contactable if anyone had any questions for another. We used Facebook primarily, for arranging meetings,

asking coding questions, and brief discussions about tasks. We did however ensure than meetings were used as the more detailed and formal approach of communicating tasks, as this way minutes could be recorded available for the client or project manager to read through.

When it came to understanding the requirements, we held a meeting as a group to discuss and question any specific details. Once this was complete, we then held a QA session with the client to clarify the questions we had made. I think that this was an important session to hold, as it then shaped the tasks we created for each sprint, and meant we could assign effort points to each task.

In order to start implementation, we had to select both a version control system, and select and IDE for the development of the Java application. When it came to the version control system, other than one member, we had all used Git before, so voted that we would use this. For the member who had not previously used it, other members from the group helped in giving basic uses, and key commands that the member would require to create branches, or add changes to the repository. Each group member helped out others when it came to issues with Git, or conflicts. If this would not have happened, the project data could have become broken, or confusing.

Selecting the IDE for Java (.NET was all shipped and ready with Visual studio) was also very easy, as we had already been using NetBeans for the tutorials, and in lectures. Each member had good experience with the IDE, though setting up the web project was a difficulty we came across. This did take time out of implementation, though was resolved, and would make a project like this easier to start should it ever again.

After the first few weeks of implementation, we had a good understanding of the scrum methodology, and the workflow was quite mechanical. Before that however, the work set in each sprint was uncomplete and pushed back to following sprints. This was mainly because of a lacking of understanding of the technologies being used. After a switch of database mechanism for the .NET project, this also incurred a setback for some members whose machines were having issues.

After issues were resolved, and each members machines were made sure that they worked the same as others (We held a few sessions to help each other), then working on tasks fell much smoother. As we did not anticipate the amount of time to get a consistent set up between machines, our time was behind at the start, and this was explained to the project manager in each

scrum review meeting.

Alongside getting the setup of each system working the same on each machine, we also made sure that when a new task was completed and pushed to the common branch that the system additions worked the same for each member. Where this was not the case, each member put in an equal effort to help others resolve their issue. Usually, the person to push the change helped to get it working for the member with issues, as they generally would have more knowledge of it.

When it came to having meetings at the end and start of each sprint review, we did make sure that they happened as scheduled, though some run on for longer than needed. In some cases, the sprint retrospective which was supposed to conclude the end of a sprint was late due to remaining uncomplete pull request code reviews. In the first half of the implantation, we had many pull requests to review, which were then performed the night before the next sprint planning meeting. This was not the case in the latter half of the project, because then all group members ensured that pull requests would be done during the week whenever possible. By keeping on top of them, the sprint meetings went much quicker and we could concentrate more on actually looking at what tasks needed more work.

We did initially plan to create unit tests as we went through coding various parts of the application, though did get behind slightly in terms of the amount of tests we would have liked. Instead, we found ourselves writing some tests at the end of implementation. Of course, this was not preferred, and it would have been better to keep up with the testing requirements. Alongside the cucumber testing though, and also code reviews on each pull request, the testing of the application in both Java and .NET was good, and covered most if not all problematic situations.

The final aspect, and one which I feel we pulled together well to complete in good time was documentation. By assigning a section each, this cut down the time to complete the report drastically, and also meant that we had time to read through others sections for proofreading. As for the evaluation, we had a session of putting ideas down onto paper, so that the section would contain collective thoughts as required of the assignment brief. Once we noted down our thoughts for the evaluation, one member wrote this, and was proofread and amended by the others. I feel the way the document was written was very efficient contained a very good amount of detail to accompany to final application.

1.3 Conclusion

In conclusion, I think that the group project was a success, even though a few setbacks and issues arose. We managed to overcome problems as a team, and where problems were still existent, getting the client involved was reflective of what should happen in a real-life situation when creating a product. The scrum methodology worked well for us, and I would like to see it used again if a project similar to this happened. If more time was given to us in this project, I think we could have gotten to grips with the technologies and setup required to get a project like this up and running.

Personally, I have learnt a huge amount of new technologies, and though at times became stressful, the end result is a good one, and one I was very happy to give to the client.

Appendix A

Group project blog

A.1 Week 1 -10/11/15

Worked on this week

- Fitbit API connection- .NET
- Storing of Fitbit token and settings to database- .NET

The first week of group project implementation followed on from initial designs of the database and high level architecture. Following on from the first sprint meeting, I assigned myself the PBI concerning connectivity from Fitbit to our application.
In order to complete this task as efficiently as possible, I looked up the Fitbit API, checking the general workflow, and parameters required in order to perform calls to retrieve tokens and then call any methods to request information such as activity data. Initially, I chose to work on the .NET side of this task, as the included OAuth 2 library in .NET should have meant that creating connections to external API's would be relatively simple.
The first major challenge was getting the initial connection, where after around a week of the 2 week sprint this was complete. At this point I was then able to perform calls to get activity data and tokens, though only using hardcoded tokens from the API, rather than storing the tokens in the database as required.
By the end of the sprint, due to the time taken to initially connect to the API, the database items had been started, however not finished.
Due to the latency of this, it was decided that the task would be pushed back to the second sprint, though this task should be complete very early in sprint 2.

A.2 Week 2 -17/11/15

Worked on this week

- Fitbit API connection- .NET- task carried over
- Device management- .NET

Following on from the previous sprint, and due to a shorter sprint it was decided that I would only initially be assigned one new task. This was to do with allowing the user to manage and de-authorize connections of users' devices to the application. In .NET, this was relatively easy to create the front end for, where scaffolding was very helpful. After this, a new controller allowed for management of devices stored in the database.
Following the completion of another group members task (login and authorization of users), I was able to link devices to users, meaning when a different user logged in and navigated to the device management page, their own devices only would appear, notifying them if a device (Fitbit or Jawbone) was connected.
In this sprint, although both tasks assigned were complete, the Fitbit task was not complete in its entirety due to the need still for the connection in the Java application.
I have also found myself performing a good amount of code reviewing and pull request viewing, and hope to keep up the amount of contribution reviewing others' work in the following sprints.

A.3 Week 3 -24/11/15

Worked on this week

- Fitbit API connection- Java
- Device management- Java

For this weeks sprint tasks assigned to me I chose to hopefully complete the Fitbit and device management tasks in their entirety by implementing the functionality in .NET, though instead for the JavaEE project. Knowing already what credentials and requirements the Fitbit API needed, I went

straight into implementing this in the most similar means possible.
Once I started creating the methods and interface to connect to Fitbit, I did notice however that the Java library for OAuth 2.0 connections was much harder to get working, therefore I and others int eh group decided upon creating requests using the standard HTTP connection class. Up to the time that is now, an issue has occurred where using the callback code from Fitbit in order to get access and refresh tokens is not working. This has therefore held back some of the functional requirements to be completed in this sprint. Device management however has been completed successfully, with it ready to check for both Jawbone and Fitbit device management as it had been in .NET.
Although the Fitbit task could be moved to the next sprint (Sprint 4), a discussion with the product owner will be happening today in order to decide realistic ways forward with how the connection to Fitbit should continue, and what sacrifices may need to be made to ensure the project does not fall behind too much.

A.4 Week 4 -1/12/15

Worked on this week

- Cucumber testing- .NET
- Report- Overview and requirements specification

Following from the meeting with the client last week, it was decided that although Fitbit integration was not working at the end of the previous sprint, it would be acceptable to push this back to the backlog, and that it could be completed at the end of the next sprint if time allowed.
This week I primarily focused on creating and running cucumber tests for the .NET application, which should stress test and make sure the GUI functions as planned. Getting cucumber set up in visual studio was relatively straight forward, by simply installing the SpecFlow nuGet package, and then the runner for NUnit tests. After this was complete, I was then able to start creating step and feature files in order to begin going through the UI.
At this stage, I have created the majority of tests. Once .NET is complete, the same feature files will be able to be used in the JavaEE applications, with hopefully only small modifications needed in order to run the step files. Other than this, I have started and completed my allocation for the report,

by creating both the overview and updated requirements. Both of these, added up to 2000 words, the planned limit for each section. In addition, I have created the overall report layout and added this to Git to allow others to add to the report.
The next sprint will combine completing the cucumber testing, and cleaning up the application alongside implementing any missing features.

A.5 Week 5 -11/12/15

Worked on this week

- Cucumber testing
- Group report

Having completed my own code for the application, and then having finished checking others' code from various remaining pull requests, I was able to finish up the cucumber testing this week. Having thought originally that two different sets of cucumber features would be required in order to test both languages, I discovered that this would be avoidable by making the cucumber code simply run in its own project, alongside any instance of the application. Because both systems use the same ID's in the HTML GUI, it meant that the cucumber application worked for both implementations. Once I had all tests passing, and with what time was left with the project was used up, I pushed the tests to the master branch of the project.
As I created the original skeleton of the document, I then went through all the sections added by others, checking for a consistent layout. Once everything was in place, I ensured the document was compiled and ready to be submitted a day before the deadline. This meant that the group then had some time to rush in any changes if they were needed.
We decided to submit the project a day early anyway, as this gave us a little time to ensure everything was handed in correctly.