University of Glasgow | School of Computing Science

Level 3 Project Case Study Dissertation

UniCom - The Student Feedback Application

Adam Christie
Alice Ravier
Ashwin Maliampurakal
Jonas Sakalys

9 January 2009

Abstract

As students, we feel that there is a lack of communication between lecturers and students in terms of our progress during various courses. This is an area that was also identified by the client, Dr Mark Wong of the school of Social and Public Policy, and in this project our team aimed to develop an application which attempts to minimise this gap in communication and improve a students learning experience at university.

The client requested that an application is developed which allows lecturers to provide their students with feedback in different areas as they proceed through a module. This was intended to provide positive reinforcement to the student, and motivate them to improve in areas brought to their attention by the lecturer.

This dissertation details the teams journey in developing this application beginning with requirements gathering, developing a specification, the development effort itself, and finally how the application was deployed. Furthermore, the ways in which the software engineering practices taught during the PSD course were applied to this project is also discussed. In particular, the use of Agile software development is detailed extensively, in addition to how members of the group worked together in this team project to deliver a product which met the requirements of Dr Mark Wong.

1 Introduction

This paper presents a case study of "UniCom - The Student Feedback Application", a project that was developed for the third year Computer Science Team Project course. The team - ESE01 - consisted of 4 Electronic and Software Engineering students at the University of Glasgow.

The project customer was Dr Mark Wong, who requested an application to be developed to improve the student learning experience and to create a new platform for giving and receiving feedback in the Social and Public Policy undergraduate program at our university.

This dissertation explains the team's development effort as we developed this application. This includes challenges that were faced during the process as well as significant achievements that were made.

In particular, it will demonstrate how the team used Agile software development to undertake this project, as well as how other software engineering practices that were taught in Professional Software Development 3 were applied during the process.

The rest of the case study is structured as follows:

We present a case study which describes the project customer and context, as well as objectives and project state at the time of writing. This is covered in Case Study Background.

Next we cover the development effort itself, including a number of issues that arose during the project and how they were overcome. This is covered in Development.

The technologies used in the project are then discussed in Technologies, including explanations for why these technologies were used and how they were applied.

A key aspect of the development effort was the use of GitLab to maintain high quality documentation and testing of code. This is covered in The Software Process.

Working well in a team is fundamental to the successful development of a project. In Teamwork we discuss how the team assigned roles to different members, as well as challenges that were faced with Agile development.

INCLUDE SOMETHING IN SECTION 7 ?

Finally, we present our conclusions of the project, including the entire process and future steps that can be taken for the application in Conclusions.

2 Case Study Background

This should include a description of the project customer (what was the nature of the organisation you were working for), their objectives for the project, and a summary of what was actually achieved. Where appropriate, this section should also make reference to similar related projects in order to make the context clear (approximately 1-3 pages).

Include details of

- The customer organisation and background.
- The rationale and initial objectives for the project.
- The final software was delivered for the customer.

2.1 The Customer

Who is the customer? Why did he want this product?

2.2 Objectives

What were the main objectives of the product? How were these documented?

2.3 Requirements

What were some of the initial requirements gathered? How were these documented?

2.4 The Product

What was the final product delivered? Did it match expectations?

3 Development

Several sections that reflect on your experiences during the team project. Each section should discuss one theme, characterised by incidents or events that occurred during the team course of the project from which you learned (approximately 8-10 pages).

Reflecting on your practice is the hardest part of writing the dissertation, so you are encouraged to talk to the course coordinators and demonstrators to find out what you could include in this section. A good source of examples of incidents for reflection is often the documentation from your retrospectives, because you used the retrospectives to identify areas of your process that could be changed or done better. You should also, try to relate your experiences to other studies available in the software engineering literature (the recommended reading is a good starting point for this). For example, if you found that you had to drop a feature during an iteration, discuss the reasons why the feature had to be dropped. Had you given yourselves too much work? Was the feature harder to implement than you realised? Had you got your priorities wrong? Then consider looking at the literature (see the recommended reading for PSD3) on project planning and estimation. Was your experience typical of a software project? What steps do other developers advocate for improving estimation? Alternatively, did you have to make some big design decisions or choice of software platforms early on in the project? What impact did these choices have? Were they the right ones? How might you have improved the decision making process to reduce uncertainty? Did you implement a prototype before proceeding to far with the main implementation? How much effort did this involve? What did you learn about the platform as a result?

3.1 The Plan

What was the initial course of action? How was the plan documented and was it realistic?

3.2 Features

What were the key features of the product?

3.3 Refining the Product

How were requirements refined with the customer? What changes had to be made to the application?

3.4 The Final Product

How was the final product delivered?

3.5 Deployment

How has the application been deployed?

4 Technologies

4.1 The Plan

What technologies were intended to be used? Was the team familiar with these technologies? Was it realistic to use them?

Could maybe include a graph of languages used

What technologies ended up being used and why?

4.2 Django

Why did the team decide to use Django? What benefits does it have? (MVC)

4.3 Bootstrap

How was bootstrap used to ensure the application was mobile compatible?

Maybe don't need this, but couldn't think what else could be used

4.4 Django REST Framework

Since we said we will be using REST to develop API, maybe say how this was intened to be used. Can also discuss how it was actually used in the application

5 The Software Process

5.1 GitLab

How was git used for version control?

What was the procedure to develop new features? (Issue, branch, code review, merge request)

5.1.1 Issues

How were issues used? Where they useful? How were tickets used?

Were new tasks created as the team discovered new tasks?

Was task meta data documented? (priority, estimates etc)

5.1.2 Branches

How was the branch made to ensure easy understanding of which issue it related to? Did members push often?

5.1.3 Commits

Where commit messages written well? Where there regular commits?

5.1.4 Merge Requests

What was the procedure for merge requests? (Template -¿ code review -¿ comment -¿ merge)

5.2 Quality Assurance

5.2.1 Testing

Did the team have an automated test suite for regression testing purposes? Did the test suite provided effective coverage of the project?

5.2.2 Continuous Integration

Was every commit tested in a continuous integration environment? Did the team immediately fix any broken builds report by the CI environment? Did the team perform code reviews (evidence from wiki or commit messages?)

5.3 Iterations and Retrospectives

Did the team hold regular post-iteration retrospectives? Were the retrospectives used effectively to (a) identify problems and (b) identify practical solutions? Were the results of the retrospectives documented? Were the solutions implemented? Was the effectiveness of a solution evaluated?

6 Teamwork

How did everyone work as a team?

6.1 Team Roles

What were the various roles? How were these allocated?

6.2 Communication

Was there efficient communication? How did the team communicate?

6.3 What could have been done better?

7 Add something here

8 Conclusions

A conclusion that draws general and wider lessons from the case study (approximately 1-2 pages).

Explain the wider lessons that you learned about software engineering, based on the specific issues discussed in previous sections. Reflect on the extent to which these lessons could be generalised to other types of software project. Relate the wider lessons to others reported in case studies in the software engineering literature.

References