University of Glasgow | School of Computing Science

Level 3 Project Case Study Dissertation

ESE1 Team Project Dissertation

Agnes Ola
Ross James Gardiner
Lorenzo Roccato
Duncan Lowther
Nawaf Al Lawati

6 April 2020

Abstract

Education Use Consent

We hereby give our permission for this project to be shown to other University of Glasgow students and to be distributed in an electronic format.

1 Marking sheme - remove before submission

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?

The dissertation should be a single PDF document of a maximum of 12 pages, not including front matter and references. You must use the LaTeX template provided for the dissertation and include all the requested meta-data. The LaTeX source for the template, your dissertation and any associated figures should be stored in your version control repository in a clearly indicated directory or branch. It must be possible to build the dissertation PDF from this source, using an automated build script, such as the ant build script provided.

- Presentation/structure - Is the dissertation complete, well organised, clear and literate? Are there examples of spelling mistakes or poor grammar in the dissertation? Is there a clear logical argument and structure to the narrative? Are references used effectively? Are the references complete?
- Description of objectives and achievements. - Are the problem domain, scope and objectives of the project clear? Is it clear what was achieved during the project and what the key technical challenges were?
- Reflection - Are there a number of experiences/critical events discussed in the dissertation? Does each experience answer the following questions: What was the circumstances of the event? What was learned as a result of the experience? What changed in the project team (if anything) as a result of the experience? Is

the experience related to other case studies available in the software engineering literature?

2 Introduction

Software engineering

This paper presents a case study of...

The rest of the case study is structured as follows. Section **??** presents the background of the case study discussed, introduces the customer, their motivation, the goals and intentions of the project and describes the final product. Sections **??** through Section **??** discuss issues that arose during the project...

3 Case Study Background

3.1 Our client

Sports Labs Ltd. is a Scottish company based in Livingston, West Lothian. The business operates within the sports surface testing and certifying industry as a service provider to many organisations requiring accreditation for their surfaces. Sports Labs work with many standards bodies such as the Fédération Internationale de Football Association (FIFA), International Hockey Federation (FIH) and World Rugby (WR) to certify more than 800 sports fields yearly in locations all over the globe.
In addition to field testing, the business operates as a consultancy, a research and development team and a testing laboratory.
Our contact within the company, Chris Dyson is current manager for the company's R&D team. He manages engineers from many disciplines (electrical, mechanical, biomedical) with a focus on developing apparatus test methods. In addition to this, Chris is involved in industrial projects with several universities besides the University of Glasgow including Edinburgh Napier and Glasgow Strathclyde.

3.2 Our project

We worked to improve one of the methods SportLabs uses to ceritify sports surfaces - Determination of Shock Absorbtion (FIFA test method 04A). It consists of an accelerometer attached to a 20kg weight, held up by an electromagnet, which is repeatedly dropped on a surface and the accelerometer data recorded. The shock absorption is calculated by comparing the maximum force on the test specimen with the reference force of impact on concrete[**?**], in other words how soft the surface is compared to concrete after repeated compression.
The test apparatus consists of a tripod, an electromagnet, a weight, an accelerometer,

a data acquisition unit (DAQ), a battery pack and software running on a Windows laptop with USB connection. The client is looking to make the test rig wireless to ease test execution, improve the DAQ (redesign the PCB to be more robust and compact) as well as get the test software running on Android so the technicians wouldn't have to carry a laptop. These changes were quickly established to be out of scope of our project; however, we worked with these future directions in mind.
We agreed to redesign the legacy software solution. Our minimum viable product consited of a program that runs on Windows, receives data from the DAQ over USB, receives calibration offset values, calculates the shock absobtion and displays the results on a graph. This program was to be written in Java for future cross platform compatibility. Our stretch goals were to implement wireless data transfer and port the application to Android platform.

3.3 Our software

4 Reflections

4.1 Windowbuilder themed part, not sure about name yet

Incidents and events: - Moving things about in the GUI disaasociated comments from corresponding code - Made merge impossible - Propagation of early, "temporary" design decisions - Difficult to rename variables - Look into why diff was so confused - Mention abandoned branch merges - Could only complete changes on the last day afer everyone else was done - Perhaps something on deadlocks waiting for others to finish

What we learned: - Make more files - Balance workload - Push earlier releases to see all the cracks in the seams - Acknowledge that there is no such thing as temporary code.

4.2 Testing or lack thereof

Incidents and events - Set up yaml / autotesting - it mostly failed due to forgetting to add supporitng files to repo. Medium useful, but mostly a dry run learning exercise - Old software was kind of shit at working without the hardware and it was hard to see if the results were the same - Very hard to make progress away from St Alwin building because you don't know if you're going in the right direction - therefore progress somewhat sporadic - Even when things do work it's difficult to prove (like with the testing exercise)

What we learned: - A hardware emulator would have helped, but would the effort of building one be worth it? - Agree on more rigorous manual tests? - Agree each person goes in each weekday and tests everyone's progress? Would have helped with familiarity of each other's code too.

4.3 Issue3

5 Conclusions

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.