CPD Report

Semester 1 Report

1605629

December 8, 2016

1 Introduction

During the first semester I have encountered many areas of the course that I need to improve on. Although I have managed to improve many of these areas, there are still a few that require more work to overcome. These problematic areas cover all aspects of the course and show a need for me to spend more time within each area, making sure I have a full understanding of them. There are, however, some aspects that over arch all areas of the course and that affect me personally, such as prioritisation. In this report I will expand on these issues in more detail, and I will explain how I have overcome them, or how I am currently overcoming them. I will then come to a conclusion about how they well make me a better computing specialist and how these skills can help me further my development.

2 Challenging Areas

2.1 Github Workflow and Version Control

The area of challenge that I met first was learning the workflow of Github in week 4 of the semester. This has proven to be a critical part of the course as many of the assignments we have been set have been group projects in which version control plays a massive part. This area has proven to be a difficult one for me as it was a completely new idea. I had not been taught anything like this workflow previously so I struggled to understand the concept and process of what must be done to achieve good version control. This skill is critical for any computing professional as you will often find yourself working in large teams working on the same product. This calls for version control to ensure that each team member is working on the same version of the software, which removes any chance of team members working on obsolete builds. However, this is an easy issue to overcome, as I feel that the more I use version control and the larger teams that i work in, the quicker I will understand the workflow of Github and other version control software.

2.2 Prioritisation

Another area of challenge that I met was learning how prioritise whcih assignments I needed to work on first and which assignments I needed to spend the most time on. This point became more apparent the further through the semester I got. At the start deadlines were fairly sparce, giving me time to work on those assignments as much as I needed, however, as the semester progressed, the time between deadlines decreased to the point of having multiple deaddlines on the same day. This required me to prioritise which assignments I needed to spend the most time on and which assignments had to be completed before others. I found this quite difficult, as I often enjoyed the work with low priority so ended working on that instead of the work I needed to complete. For

example, I spent time making a working prototype for the game we are making next semester, and in turn spent less time on the assignments that are due in at the end of this semester. Work prioritisation is a key skill for any job. It is especially important to computing specialists as they often need to prioritise work for different features based on how integral these features are to the end product. i feel like having this experience of struggling to prioritise will, in itself, help me to improve my prioritisation skills as I will learn through experience the advantages of prioritising work that is critical to the success of a project.

2.3 Programming for Mediums other than Visual Outputs

My third main area of challenge occured when we began our tinkering audio module. The main issue with this was that I had never programmed for any output other than a visual one. That is to say that I was used to changing graphics and seeing a visual change. So i found it very hard to visualise the wave of sound I was creating and judging what changing parts of the code did. Knowing a broad spectrum of outputs is an invaluable skill for computing specialists as it allows them to be a more valuable member of any team by making them a more versatile programmer overall. I feel like I struggled with this as I had not had any previous experience in audio programming, and so I couldn't imagine how to visualise the wave while I was programming. I feel like as I spent more time working on programming audio I was able to visualise the sound wave more clearly and began to gain a greater knowledge about what in the code caused different effects in the wave. I feel like this experience with audio engineering has been invaluable in opening my eyes to the wider world of programming.

2.4 Agile

I think that another area I struggled on was implementing the Agile principles into the way I worked in a team. I gained a strong knowledge of the Agile principles themselves

during the lectures we had on them, however I found that actually implementing them as a team was significantly harder than I had first anticipated. I felt that the hardest part of thisprocess was ensuring that daily stand-ups were as quick and clear as they could be and that any blockers that were reported were actually removed. I overcame these problems by really cutting down the detail I spoke about my points in, while still getting across the critical points. I feel that I still have a way to go in incorporating Agile into the way I work completely, but I am at a stage where it is improve my productivity and efficiency within a team medium. Having a strong knowledge of Agile and experience in working in Agile teams is a key skill for computing specialists as the vast majority of development teams use Agile and thus, to be an effective member of any team, you must know how to fit into the process seamlessly.

2.5 Planning

A major area of chalenge that I encountered was planning my code. I first encountered this when I had to create psuedocode, which I struggled to do, as I found that I couldn't effectively do this without having any actual code infront of my to try out and get working. I think this is a result of how I first began to learn programming, which was self taught. So I missed many good practices such as this. the act of planning out your code before you begin programming is a very useful tool for computing specialists as it reduces the amount of errors within your code, and thus reduces total development time. It also allows you to ensure that your code is as clean and refined as it can be which helps massively with its maintainability. I overcame this by practicing psuedocode for all the code I started. This got me intot eh habit of planning out my code. I saw a significant increase in the maintainability of my code, and it made my code significantly easier to read. This had a major knock on effect during my group work as it ensured that the other group members could read and adjust my code with ease.

3 Conclusion

I feel like I have seen an improvement in my skills as an overall programmer during this semester and I belive it is mainly down to the effort I put into trying to overcome the areas of challenge mentioned above. Although, that in no way means that I should stop working on improving these areas. In fact i means I should put more effort into it, so I can see an even greater improvement in my skills. I believe improving these skills will not only make me a better individual programmer, but also a better group programmer. This will make me a better computing specialist in the long run, and more appealing to possible employers.