

## Introduction

For the industrial experience component of my BSEng degree I worked for the XYZ on two projects. The first project was actually done for the COMP4510/COMP4520 units and involved the development of a command stack verification system for the OCC (Operations Control Centre) software of the ABC project.

## Learning How To Learn

I think I will start with 'learning how to learn'. It's quite apparent that the software industry will continue to change and grow, technology will continue to improve, and new tools will be developed to aid in the development of software. As such, it is an important aspect of being a software developer that one can change and grow with the industry in which one works.

In first year we were taught programming. Those lessons just happened to be in Modular 2. In later years, though, we were only introduced to other languages (most notably C and Java), and were expected to learn the rest ourselves. What I took from this experience was the ability to learn a new programming language myself, and because I had been taught the fundamental principles of programming, instead of any particular programming language, I was able to do that. This became more apparent, though, when I started working on the OCC, as it was developed in Visual Basic; a programming environment that I had never used before. After the few years of programming with Java, it came as quite a shock that Visual Basic did not have inheritance, even though it did have classes. Despite this, I was able to employ my knowledge of programming paradigms to this “new” language, and produce just as well structured code as I could with any other language I was already familiar with.

Source code control is another area where learning the fundamentals was more important than any one particular technology. We were introduced to RCS in second year, left to teach ourselves about CVS in third year, and then I was required to learn Visual Source Safe for the OCC source code control. Now, in my opinion CVS is a far superior source control system than VSS but the basic operations are covered, and apart from some annoying crashes, it worked well for a simple project like the OCC.

## Process (or the lack thereof)

Since the very beginning of my degree, lecturers have been telling stories of projects gone bad. Although I did believe the stories were true, I still struggled to see how someone could stuff things up so much, that a project would be canceled completely. I think it's just something you have to see, and experience, for yourself to truly understand the situation.

Although the OCC project wasn't canceled, I do think I got to experience a project heading for the canceled heap. I think one thing that kept the project going was the fact that if the project had failed then the entire satellite project would be useless. I should also like to point out at this point, that my experience in the work force was more a lesson in what *not* to do, rather than how it should be done. With that in mind I shall relate what I learned about the software development process at uni, to my experience developing software outside an academic arena.

Each document developed for the ABC project had a sign off section of the front page. Usually three signatures were required, the author, the manager of the author, and the manager of the manager. I'm not really sure how important it is to have levels of management sign off on documents, I suppose it depends on what it is the particular manager is looking for before they sign off on it. I would have thought, from my learnings at uni, that the author, the manager of the author, and a few other people that would have input on the document under consideration, would be present at some kind of review of the document. Therefore those two signatures would be there to indicate that the document had, in

fact, been reviewed, and then the manager of the manager would sign off on the document as part of the official completion of a part of the project, or even as part of a milestone.

As it turned out, that wasn't quite how things were run on the ABC project. Although there were three sign off places on the front of each document, all three were rarely used (in fact, two were rarely used). Not only that but the documents were usually signed off by the author right after they'd finished writing them.

So how does this relate directly to my work on the project? One of the tasks I was set late in the project, was to develop the ground station end of a bootloader image uploader module. It would be used for uploading an image of the on board software to be used instead of the image in flash, if the image in flash was damaged in some way. Basically a boot by telecommand facility.

Originally the large file upload module was going to be used for this, but the protocol for uploading large files was far more robust (which is to say more complex) than could be used during the boot sequence, as the bootloader code had to be stored in a small amount of ROM. So a simpler protocol was developed by Kurt, specifically for the bootloader.

Kurt developed the protocol, wrote the document, signed off on the document, designed and implemented the on board part of the system, had it burned (permanently) into ROM, and after doing all that, he finally got around to asking me to develop the OCC end of the module. "No problem" I said, as he handed me the document, but problems there were, as I later discovered.

I could probably write a masters thesis on the problems I encountered while developing the uploader, but suffice it to say, I returned to Kurt after just 30 minutes of reading the document, with a heap of questions, and the document covered in red ink. I didn't need a formal review to tell me that no one else had ever read the document, but I can say, without a doubt, that *weeks* of work would have been saved if Kurt had, at the least, worked with me on the protocol, instead of doing his side of things and then expecting me to conform to everything he'd done. Who designs a server without designing the client that will talk to it?

When studying the review process in COMP4100, and doing the review of the third year's SRSs, I got a good feel for how important reviews were, but after writing the uploader, I was thoroughly convinced of the importance of reviewing, or at the least having someone else read over, a document (or any other work product of a project), before going ahead with the development.

On to another aspect of software process. Of the people who worked on the OCC, exactly zero of them had ever developed ground control software for a ABC before. Although this can probably lead to all sorts of different problems, the main one for the OCC project was that nice-to-have features were being implemented before the core functionality was properly finished. This mainly came about, I think, because no one was quite sure what features would be absolutely necessary, and what ones were just nice-to-haves. Even so, there were still some core features that could be recognised as absolutely necessary.

What this caused, though, was a bloat, and not just in the code. Many of the design documents became much more complex and difficult to understand. The OCC became far more complex than it ever needed to be. The best example of this would be the macro module. It was thought that many of the tasks of the OCC could be automated by the controllers of the ABC, and therefore would require less human intervention, so a macro system was designed to allow the OCC to be programmed. This module soon became a behemoth module that didn't even come close to working (and to this day doesn't). Worse still was that other modules were modified to accommodate programmability, which added even more complexity. This was all going on before the core features of the software were even completed.

I would like to point out that all of this happened before I started working on the OCC, and that much of my time was spent developing and maintaining the features that were actually being used by the ABC guys for testing, i.e. the features that I would have classified as the core functionality.

I would also like to say that I didn't come to these conclusions until after I had researched my assignment on agile development practices. I believe many of the ideas that I read about could have really help the development of the OCC. In particular the short release schedule and a focus on having working features as soon as possible, would have kept things a little more on track.

## **A Clash Of Ideologies**

I thought that I had gotten a fairly accurate experience during the group project (COMP3100) and the “individual” project (COMP4510/20), of what it was like to work on a substantial project. As it turned out those were very sheltered experiences, when compared with life in the “real” world.

One of the things that differ between uni and other projects is that, outside of uni, projects can be spread over not just development groups, but over organisation, over states, and even over continents. With so many different groups involved there is bound to be clashes of ideologies.

One such clash was within the development group working on the OCC. The OCC was developed by two groups, one in Canberra, and one in Adelaide. Both groups were at the extremes of development ideologies, but unfortunately they were at opposite extremes.

The guys in Adelaide were almost cripplingly rigid in there need for documentation of every aspect of the OCC's development. Every decision required some kind of formal meeting, with minutes taken, and action list written up. There were times when I would attend a meeting where a decision about something I was working on would be made; five minutes later I would have the problem fixed; 30 minutes after that I would receive an email with an action list from the meeting with my actions detailed, it was agonising.

At the other end of the spectrum, the situation in Canberra was almost the opposite. Sometimes meetings were held without all the required people being told about it, sometimes decisions were made about the ABC that would effect the OCC, but no one told the OCC team until the OCC broke. As I've already explained many of the documents were largely useless.

As you can imagine this differing of procedure caused many a problem. As an example, Kurt required a number of problems to be fixed in the large file transfer module, which was developed by the guys in Adelaide. The problems had brought the ABC testing to a standstill and Kurt was becoming increasingly impatient. Meanwhile the guys in Adelaide were conducting meetings to discuss what the problems were, who should look into them, and who should be responsible for ultimately fixing them.

So where do I come into this? Well I had the unfortunate position of being cc'ed on all the emails sent between Kurt and the guys in Adelaide because I was a member of the OCC team. Many emails later Kurt had had enough and sent a... lets say, less than productive email, to Alistair (the guy in Adelaide) which basically said that he was letting the team down (plus a few nastier things) etc. To keep a long story short, Alistair was demanding an apology, Kurt was refusing, and I ended up in the middle of it because Kurt was now asking me to fix the large file transfer module so that he could finish his testing. There certainly wasn't anything in my software engineering degree to help me deal with this situation.

Anyway, after about an hour and a half of looking into the problem, I found that it was quite a simple error that could be fixed in no time. In my opinion the whole situation could have been avoided if someone had simply sat down to find out how big a problem there really was, instead of (Kurt) demanding it be fixed immediately, and (Alistair) calling for meetings to decide what needed to be done before even looking at the problem. For me though it was an interesting and helpful experience,

and one that you don't get from uni.

## **Conclusion**

All in all, I would say I had a very positive experience while working on the OCC, learned a lot about the more human side of software development. Having said that I hope to, one day, have an experience on a project that uses a more successful process, if only to compare it to my experience on the OCC.