Brian Breniser

CS 300 Retrospective document

Over the course of this Fall term I learned quit a few interesting things about computer science, and software engineering. The project we participated in allowed us to experience some new concepts I have not previously seen. A projects requirements, design documentation, and test documentation is an integral part of the software engineering process. Without these critical concepts our projects would fail and ultimately always end up in the scrap bin. The project we did this term was interesting from a process perspective, and will be useful in the future of my education.

Documentation of a projects work is very new for me. I think this terms project had some issues when it came to what we needed to document. I felt I didn't now exactly the extent of the documentation I needed to do; and I think I ended up doing less documentation than I really needed. In talking with my colleagues I fond most others did more documentation, although some of them still felt like they did not do enough. Our requirements document was very overview, and I felt did not have enough detail written in it. In the future when I write requirements documentation I will focus more heavily in getting the details down, instead of just the broad requirements.

For our requirements documentation, I felt that it's usefulness in understanding our project was fairly limited, since our projects size was quite limited. In theory a much larger project would benefit from much more documentation, since it would reflect the details that are harder to keep all in one place in someones head at once.

Our design document came in much more handy for myself, where I decided what my file structures would look like, and what, exactly, I would be coding. I still felt like I did not get enough details out of the documentation, and in talking with others my feelings were confirmed once again. I was, simultaneously, doing a personal project that I had done much more documentation on, but that was probably because I understood the project better.

In the future I would certainly put more details in the documentation, and have better direction on what to put in them. Unfortunately I would need to understand the project much better before doing extended documentation.

The testing document proved a little less useful, since it was truly so under prepared. I think that I had a very hard time visualizing the tests I needed to write before actually writing the code first. Test driven development is a very interesting concept indeed, but would require quite a shift in how I do my work.

I learned quite a few lessons over the course of this term. I learned it is better to be prepared, even if it isn't quite 100% prepared before the project starts, something on paper is better than nothing. Especially when it comes down to design documentation, I felt that was the most useful. I also feel that testing in general is an extremely underestimated tool. I ended up writing about as much test code (although most of that ends up being copy/paste) for testing than for executing the actual program. And although I didn't get the opportunity to test my user interface (programatically speaking), the feeling of running my tests and getting a no-news-is-good-news response was so liberating. I felt that so good that I didn't have to touch every single little thing by hand, because my tests were succeeding I knew my code was working. It ended up helping me build my user interface faster, since I didn't have to test that every function ran properly, just that my UI did what it was supposed to.