November 28, 2017

Deliverable #6 Reflections

1. SOFTWARE PLANNING

A. All three of us had prior experience with the workflow of a software engineering team with our Software Architecture and Design class the previous semester. In this class, we learned how planning is paramount in software design and that even with an amazing plan and design architecture it can change over the course of development in the software process. The same goes for testing, this aspect of the process that we have learned about this semester. We learned how important it is to maintain a functional testing framework throughout the development of the software project so that you know that the code has been well tested since creation.

B. Our group this semester suffered serious time constraints and availability in group meetings. We conquered this constraint by careful delegation of work that played to each of our individual strengths. Andrew's strength lies in scripting as he has some prior experience dealing with them. Noah's strength lies in keeping dependency directories in check as well as keeping up with customer relations. Brian's strength lied in structure and making sure everyone was up to date with ongoing deadlines and ensuring that certain frameworks upheld the customer's expectation.

2. CONSTANT CUSTOMER INTERACTION

A. Having customer (Dr. Bowring) interaction throughout the development of the testing framework was very helpful in ensuring that we were developing exactly what the customer wanted. At first we struggled to deliver what the customer wanted because we did not communicate as well as we should have. After our first deliverable we made sure we knew exactly what the customer wanted to ensure that we did not lose the contract.

B. Customer interactions are vital to a software project, even in the testing framework. Without this you could proceed through the planning process without ensuring that the code performs up to the customer's standards. Before this project, we had somewhat limited experience with this; moving forward we all three now realize the paramount importance of open communication with the customer.

3. BASH SCRIPTING

A. In the college's computer science curriculum there is only one class in which we have worked with a Linux platform and used bash scripting. That being said, it was

very limited as most of the class was limited to C programming using bash to compile and run our programs. Needless to say that this project was the most bash programming that any of us had ever experience and it was a great learning experience. From generic script creation to creating a script that will compile and test Java code, we have tackled bash scripting thanks to this project.

4. OPEN-SOURCE QUALITY

A. This project has opened our eyes to the openness of open-source software. This was most ours first experience with open-source software and we didn't know what to expect. Our first troubles experience with the Sigmah project were the instruction sets provided. We found 3 different sets of instructions, one for users, one for administrators, and one for developers. On the surface this would seem to make sense, but after proceeding with the developer instruction set, we saw how poorly crafted it was. On one seamless document of instructions for Mac, Linux, and Windows users, there were links for specific sets of the instructions that were "uniform" for all three platforms that linked back to the Windows set of instructions. This was confusing at first and caused a bit of delay while setting up the Sigmah software.

B. Our second trouble was in the high number of dependencies in the Sigmah source code. While this created a serious pain for us while compiling our script and ensuring that our class paths were correct, one would beg the question: Why so many? We understand that open-source software is just that, open-source but there were hundreds of dependencies scattered throughout the source files and after a while it just seemed entirely unnecessary and honestly, a bit rushed and poorly done.

5. CONCLUSION

A. To be completely honest, we had no idea what a testing framework was until we took on this project. After learning about the concepts of a testing framework, and what test cases and drivers actually were, things started to make sense. We all three now feel comfortable saying that if required to do so by our jobs we could implement a testing framework and test a software proficiently. Our experiences and conquered troubles in this project have more than qualified us as Testing Engineers.