Reflective Journal B

Name: Marc Tibbs
Email: tibbsm@oregonstate.edu
Student ID: 933270515
Class: CS361 Software Engineering – Winter 2019
Assignment: Reflective Journal B

Thursday, January 31, 2019

I've been meaning to start this journal for the past couple of days, but I've been bogged down with other things. For one I've been busy with the other class I'm taking this winter, CS344. Our new assignment has just been released for that class, so I've been spending my time mostly working on that. On top of that, there are interviews and coding tests for internships are starting to come in for this spring and summer. As important as school is to me, I prioritize work or the opportunity for work/experience over school. I believe I will still have enough time to finish everything and still get good grades, but at this point it feels as if everything is coming at me all at once. So, I haven't been able to commit too much time to this class as of late, but after a few more days of grinding I should be back on the radar for the class.

Sunday, February 10, 2019

Alright, so I've really been lagging behind on writing in this journal this time around. I, of course, have my reasons, but considering that I will have to turn this journal in in a couple of weeks, I really need to get back on track. I mentioned a little bit in my last journal entry that I have been busy with career related activities as of late and I think I might of underestimated the time that I would need to allocate to those activities. I received a quite extensive coding assignment from a potential employer and I ended up spending the past week mainly working on that assignment. After that was completed and turned in I had to turn my attention to my class assignments that I need to turn in next. Luckily, I have so far been able to catch up and stay on top of things. I have a few more assignments that I need to finish up for this week, but after that I think I'll be back on track. Expect me to write another journal entry tomorrow! No more slacking off on this class.

Monday, February 11, 2019

- 5.1 What are the advantages to having documents for the requirements and design?

As always the advantage to having documents for requirements and design is the ease for others to read and understand why certain requirements and design decisions were made. Without this documentation projects can be a black box and very hard to figure out for any collaborators or mentors that are trying to improve the project.

On the other hand, documentation also helps designers and developers find any holes in their design. I find that when I have to explain things out in documentation I find out that I missed something in a project.

Tuesday, February 12, 2019

- 5.2 What are characteristics of a development project (i.e. size, complexity) that would justify creating these documents, in advance of coding?

I think the larger the project is the more effort that needs to go into the documentation of the project before coding anything. This will save the developers from making many mistakes, which will be sussed out in writing the documentation in advance. If you're creating a small enough project then it doesn't really matter as much. Problems can be solved rather quickly and anyone else can read through the code and understand it. However, when you have some monolithic project on the horizon, it is much easier for everyone to break down the large project into smaller chunks and tackle the smaller bite-sized issues that way.

Wednesday, February 13, 2019

- 5.3 We will be moving on to agile methods, where documentation is de-emphasized. What are circumstances where this might be acceptable?

As mentioned in my last post I believe that agile methods are better suited for smaller projects or projects that are implemented with iteratively. The agile method is suited for this because it emphasizes the need for steady, iterative improvements to a project vs.

the iterative approach. Once a project becomes larger, it becomes necessary to think in pieces of a whole rather than continually juggling with the whole system every time you sit down to do some work. I think that some projects benefit from a dual approach of using agile and waterfall methods. The system as a whole usually needs to be dealt with on an incremental basis, but the sub-projects can often be tackled with an agile approach.

Thursday, February 14, 2019

- Lecture 5.1 Object-oriented Design Overview

I just watched lecture 5.1 on Object-oriented design. I really enjoyed the lecture and it was a great refresher on OO design, which most classes have not touched since the intro classes. I think the principles and rules that come along with good implementation of OO classes are easy to forget. We often write code that is sub-par when it comes to maintainability and in the end it hurts the project as we are more likely to not want to go back and work on the project.

Friday, February 15, 2019

- Lecture 5.2 Design Patterns I

Today, I watched lecture 5.2, which covered around half of the design patterns that we will cover in this course. It was a really great lecture and I was happy to see a lot of patterns that I have been seeing used at my internships. The first time I heard about the design patterns at work, I was completely lost and had to scrounge the internet to make sense of the things that my co-workers were talking about. With time I learned more about a few of the design patterns such as adapters, observables, and builders. I'm happy to learn about the other ones as well since I'm sure that they will come up sooner or later at work. The lecture did a great job giving the basic overview of what the design patterns are and what they are used for. I will have to do a little research on my own to find out more details on the patterns.

Saturday, February 16, 2019

- Lecture 5.3 Design Patterns II

I watched the second part of the design patterns lectures today. It was a great follow up the last lecture and finished off introducing the design patterns that we will be following in the course. Of these patterns I had only seen the factory design pattern before, so there was a lot that was new for me. I think just like the last lecture I'll have to do a lot of my own research to find out more about these patterns. And I think implementing some of my own will really help out in understanding them.

- 6.1 Regardless of the method, why are requirements and design documents essential?

Regardless of methods, I think that requirements and design documents are essential because they give a clear, actionable direction for the project. Without these types of documentation, a project can waver all over the place and not really get anywhere and even end up doing the same work over again because of bad planning. I believe that documentation like this is absolutely the bare minimum for any serious project. Moreover, they do not take too much time especially when you consider how much time and effort you save with some simple plans.

Sunday, February 17, 2019

- 6.2 What are characteristics of a development project (i.e. size, complexity) that would justify using agile methods?

I think I went over this in one of my earlier entries, but just to reiterate I think that agile methods are better suited for smaller, less complex projects. The benefit of using agile methods is the speed in which you can accomplish project goals. However, for larger, more complex projects it doesn't matter how fast you can implement things if you have a poorly designed plan for the project. In the case of larger projects, I believe it is better to break up the project into smaller sub-projects and then using the agile method on those smaller projects.

Monday, February 18, 2019

6.3 True object-oriented programming applies discipline to the development process. How can this create more reliable or trustworthy software?

True object-oriented programming come with many necessary and helpful rules when it coming to implementation. By its very nature OO programming has the developers make strict rules about how they are going to structure their software. While flexibility in code often makes it easier to code projects, it also ends up making messier and harder to maintain code in the long run.