Brady Field

First Design Attempts

**What is the primary reason why you won't get it right the first time? Clearly state the reason and discuss the reason.**

We don’t perfectly understand the problem. We also do not understand the possibilities. Just like there is no perfect solution in coding, there is no perfect solution in design either. We are trying to find an optimal design with the limited resources we have. To complicate things further, we are trying to uncover the problem. We need to dig with our stakeholders to attempt to define what the problem is so that we can properly understand a solution. This takes a lot of work. We can’t even get the problem right the first time, so we should expect to never get the solution right the first time either.

**Is this (not getting the design right the first time) a bad thing? Clearly state "yes" or "no"; don't be subtle. Why or why not?**

No, this is not a bad thing. We learn from our mistakes. The phrase from Tom Kelley in the Deep Dive video is “Fail often, to succeed sooner”. Not only does this help build the right solution, it also teaches us valuable lessons along the way. We can learn to avoid the pitfalls. We can share what we learn to help others avoid the pitfalls. When learning about programming as beginning students, we often made many mistakes. This wasn’t a bad thing because we learned how to fix these mistakes. The more mistakes we made, the more we learned how to fix. We learned a lot in class, and while we had a lot of help getting pushed in the right direction from in class activities, it was fixing our mistakes in the lab that really helped us become better software engineers. I expect the same is true of design. After going through the pain of fixing things, we will be less inclined to make the same mistakes in the future.

**What is the primary thing you can do to reduce the time and effort necessary to get it right? Clearly state the primary thing you can do and discuss it.**

From working on the SRS I have learned that the big challenge is working with all the different people involved. We have been taught repeatedly that Software Engineering is not about technology. It’s about people. We have a lot of great minds to work with. These minds are all necessary because the system will most likely always be too complicated and/or too large for any one person to understand. If all these minds can cooperate and be utilized properly, then we can drastically reduce the time and bloodshed it will take to get the right solution. We must learn to work well with others so that we don’t get in our own way.

**What is the best way to know you have it right? Clearly state the best way to know you have it right and discuss it**

The best way I can think of to verify that you have it right is when the stakeholders say that it is right. The stakeholders are the ones with the need. If they do not say that they are satisfied, then we have failed. We cannot know that we have built the right system until they have told us that we have. We can build the system right and work out all the bugs we can find, but if we haven’t built the right system, then we have failed. That system will be a waste. This is why it is important to involve the stakeholders from the very beginning. They know what their needs are. They are the ones who define the problem, and they are the ones who sign off on the solution.