# **Design Review Document**

## ***WithU***

**Team:** Sam Patterson, Joshua Chaimson, Pritesh Kalantri, Will Roberts, Mingren Shen, Shantanu Singhal, Jason Waner

### **Design Review for *Launch Vehicle Ground Support System***

### ***<2017-10-20>*, version 2.0**

**Document Revision History**

### *Rev. 1.0 <2017-10-16>: initial version*

### *Rev. 1.5 <2017-10-18>: more detailed version*

### *Rev. 2.0<2017-10-20>: final version*

## **Summary**

The biggest points from the design review are that some areas need to be laid out in a way that someone who is not in your group can come in and start to work on the project. Many of the diagrams were confusing, and some areas were missing a visual components where visuals should’ve been. One area where we continually pointed out was that the Design and Specification document needed a UML diagram to make it less confusing. Not only will this help out your team, but will help out your customer see what you are working on. Another area we talked about a lot was can this project be completed in the timeline we have for this class. Most of the implementation cases were for iteration 1 and only a couple for iteration 2. This could cause problems if your difficult tasks don’t get finished in a reasonable time, and you can’t proceed to your next tasks. Overall as a group we liked how specific, consistent, and precise your designs are. This will help you design and implement great software for your customer.

## 

## **Questions**

1. **Are there any inconsistencies in the design? That is, does the document contradict itself?**

When comparing the consistency between the Requirements and Specifications document to the Design and Planning document, they both appear to complement one another. This being said, the Design and Planning does not appear to contradict itself. However, the lack of a UML diagram does make this document a little harder to follow. When having a project of such detail as Wisconsin Propulsion Lab Ground Support System, it gets hard to follow all these details when they are all text. Throughout the document, specifically the Design Details section, it was easy to forget with class was in charge of certain features. With this, some inconsistencies could easily go overlooked. Certainly, a UML diagram would have cleared up this confusion and made the whole flow of the document much more simplistic.

1. **Are there omissions in the design? That is, are there elements that are mentioned but never discussed, or obvious pieces that are missing**

When looking for elements that are missing, there is nothing that really catches any attention. The document is consistent, but does not elaborate on majority of their elements. When looking over the System Architecture, most of the section is fairly brief and does not go into much detail. Even when referring back to the Requirements and Specifications document, the system architecture lacks detail. If the group were to elaborate more on each of the key components that make up their system, the documents, and even the project as a whole, would give the reader a much higher level of clarity and understanding. With this, the Design Details sections is well explained. The group did a good job with briefly explaining what each attribute, method, and event does for each class. However, as stated for the previous question, this section would be more clear if a UML diagram were present.

1. **Are any parts of the design unclear? The standard should be that given the design document, a competent programmer can code the project. Note that “clear” does not mean that the document must be very detailed. We assume that a decent computer scientist can fill in missing details, provided the overall document is clear enough.**

Overall the design was fairly clear; however, there were a few recommendations/areas that could use a little more explaining. One of those being when talking about the model to include more visuals to help the reader follow the backend design more clearly. The detail in the attributes/methods is great, but including a visual of what makes up your model and the relationships within would go a long way in helping others see the relationships better as well as it be a nice thing for all members of your team to be able to look back to rather than have to sort through all of those lines of text. Another area that was a little unclear was the dependencies listed in the implementation plan cases. For instance, multiple cases had “Configuration Service” listed as a dependency, but there was nothing on the implementation plan about creating this. If this configuration service was already completed outside of the scope of the project, then maybe just explain how it is dependant on that service, however the service is already completed. There were several other dependencies like this which appear in the system architecture part, but not listed as something to be implemented. That would help clear up whether that was something missed and not listed or simply was already done and just works off of that. For other dependencies listed on your implementation plan you have a matching implementation plan that goes with it beforehand such as “Sequence View” or “Live Graph View” so those make sense, just unclear about those not listed, that’s all. Other than that, the design was pretty easy to follow along with and looked good, besides a few minor recommendations.

1. **Are there technical errors in the design? Is there any statement of fact that you know is false?**

There are no obvious technical errors in the design. Having no knowledge of programming with an Arduino, it is difficult to determine if there will be any difficulties pulling data from the device at specific intervals, or pulling data from the device in general. One aspect that wasn’t mentioned in the design document was what different programming languages will be used. Working with multiple languages between different systems could be challenging and lead to some technical difficulties further along in development.

1. **Has thought been given to testing? How would you test this design? What, if anything, could be done to make the design easier to test?**

After reading through the test section of their design, it looks like a lot of thought has gone into the testing your software. Details for how specific areas would be tested are specifically written for the unit, integration, system, and regression testing. It is great to see what specific inputs and outputs your group needs to use and get to make testing as easy as possible for you. One way you could make your design easier to test is used in the built in testing functionalities of your IDE if you are using one. After reading over your papers, I didn’t read anything about what IDE you are using, but if you are using an IDE like visual studio, there are many features that can help you with testing your software. For example, in visual studio you can write, execute, and debug all of your unit tests. If I were you, I would use what the IDE provides to help you develop your software.

1. **Does the design make realistic assumptions about the environment? That is, will the team have trouble getting access to important external components (e.g., specialized hardware) and are the systems the project needs to interact with suited to the purpose?**

The design does make realistic assumptions about the environment. The only area where I can see a problem is not accessing the actual liquid rocket engine, but everything else though seems it has been thoroughly thought out. The two specific components that your team needs are Labjack T7 Module and Arduino/XBEE Telemetry Modules. After reading your papers, these two components are necessary to pass data and gather information from the fuel cell to the user. In the customer section of the paper it talks about about WPL students being able to test out the system using these components, so it does not seem that you will have a problem accessing these components to use for testing.

1. **Does the plan seem realistic? Are tasks at a reasonable level of granularity and is it clear what each task means? Do the time estimates seem appropriate? Do any parts of the plan seem risky in the sense that they are likely to become a bottleneck to further progress?**

While tasks are at a reasonable level of granularity and most of the descriptions for the tasks are clear and suffice, there is some worry as to the iteration plans being realistic. Only one of the implementation plan cases (Creating the composite live view) is listed as being a part of iteration 2. Everything else on both the frontend and backend is listed as being a part of the 1st iteration, which seems overly ambitious. The time estimates themselves seem appropriate, but again with iteration 1 being due in only a few weeks and several tasks receiving moderate difficulty rating and multiple week time estimates, the plan still seems unrealistic. Splitting this plan into two iterations, if possible, will help make this project more realistic and prevent errors from occurring while having so many separate tasks trying to be completed simultaneously in such a short period of time. One area where I think could cause problems is that if you are not able to develop a way to get the information from the arduino to the user. If this task doesn’t get implemented, it looks like other areas of the project can’t proceed.

1. **Any other comments?**

Combining hardware and software like your project is alway a hard task because you need to be strong at programming, creating a MVC model, and understanding the technology details in hardware like ​Arduino​ ​UNO and LabJack API. But as a group, we think this is a good idea. We fully appreciate that your group is willing to present this wonderful project to the class. We believe that if there were more figures and background knowledge explanations, they would greatly improve our review of your project.