Assignment 3 Results

Erik Orlowski

Comments on the Design

The design certainly made my implementation a lot easier. The biggest way was in figuring out how to implement the observer pattern to interact between the Model Service and Controller Service. Taking a top-down look at this ended up in a very clean and elegant interaction between the services that likely wouldn't have been possible without this approach.

To improve the design, I could have been clearer about how outside stimuli would be interpreted by devices. This would have made the design and implementation for stimuli such as camera events and Ava voice commands much simpler. In a larger sense, when dealing with partially defined architectures and systems, I've learned the value of making decisions early on behaviors and interactions, either myself or with a team to drive the system design forward.

Integrating the Model and Controller Services

As I commented above, the main difficulty in integrating the two services was understanding how the model service would respond to outside stimuli. After this, the model service sending information to the controller service became simple with the observer pattern. The controller service requesting information from the model service was somewhat more difficult, because the commands it was parsing from were meant for human and not machine consumption. If I had known about this future use case when implementing the model service, I likely would have made different design choices for configuration command output.

Design Review

In the design review, my partners gave me feedback on the readability of my class diagram. Through this, I discovered the ability to manipulate the diagram in Plant UML by changing the line lengths of relationships.

I gave my partners feedback on their sequence diagrams where they would be helped by adding instance specific, rather than just class specific information to these diagrams. I also gave some feedback on how the Controller Service should interact with the Model Service in one of my review partner's class diagrams.

Changes to Design

Initially, I was under the belief that the observer pattern would only be able to handle some of the rules for the controller service and I would need to add an additional mechanism, such as a CommandFactory to handle additional input. However, as I implemented the solution, I realized that the camera and ava commands I was expecting to come separately from the model service could instead be processed as statuses and values and therefore take advantage of the observer pattern.

This eliminated the need for a CommandFactory and greatly simplified the interactions between the Controller and Model Services.