Sprint 4 Report

Team Members: Austin Wentz and Jordan Doell
Date: February 8, 2013
Class: Senior Design
Subject: Sprint 4 Report
Sponsor: L-3: June Alexander-Knight

Backlog

Completed

- Configure EC2 server to act as middleman between iPhone app and xmas lights
- Develop RESTful web service to allow iPhone to send commands and Raspberry Pi to get commands
- Begin connecting iOS framework to UI

Remaining

- Develop and implement iPhone app to interface with controller to control lights and music
- Program and configure Raspberry Pi to act as midi sequencer for lights
- Implement JSON-RPC on Raspberry Pi
- Implement JSON-RPC on EC2 server
- Connect iOS framework to UI
- send JSON from iPhone to server

Christmas Light Controller Progress

Austin Wentz

Configuring EC2 Server as "Middleman"

To avoid issues with firewalls, network configuration, and etc. we decided to use a server as a middleman for communication between the iPhone and the Raspberry Pi. We went with an Amazon EC2 server running Ubuntu. When the user wishes to send or receive information, the iOS app will send notifications or requests to the EC2 server. The Raspberry Pi will run an application which will periodically query the EC2 server for any new information. Figure 1 shows a diagram of this configuration.

Figure 1

Define Interactive Lighting JSON-RPC Interface

JSON-RPC is a remote procedure call protocol encoded in JSON. It is a very simple protocol, defining only a handful of data types and commands. JSON-RPC allows for notifications (info sent to the server that does not require a response) and for multiple calls to be sent to the server which may be answered out of order.

JSON-RPC will be used to send commands to the lights and to retrieve lists of songs and light sequences available to be played. The interface is defined as follows:

```
//play the song identified by songId
//json example: {"jsonrpc": "2.0", "method": "playMusic", "params": [3]}
void playMusic (int songId)

//run the light sequence identified by lightId
//json example: {"jsonrpc": "2.0", "method": "runLights", "params": [2]}
void runLights (int lightId)

//play the song identified by songId and run the light sequence identified by lightId
//json example: {"jsonrpc": "2.0", "method": "playMusicWithLights", "params": [3,2]}
void playMusicWithLights (int songId, int lightId)

//request a list of valid songs and songId's
//json example: {"jsonrpc": "2.0", "method": "getMusicList", "params": [], "id": 1}
char* getMusicList ()

//request a list of valid light sequences and lightId's
//json example: {"jsonrpc": "2.0", "method": "getLightList", "params": [], "id": 2}
char* getLightList ()

//request a list of valid song/light combinations
//json example: {"jsonrpc": "2.0", "method": "getMusicLightList", "params": [], "id": 3}
char* getMusicLightList ()
```

Developing RESTful Web Service

A web application/service is now needed to run on the EC2 server. The service needs to perform two main tasks: accept new information from the iOS application and allow the Raspberry Pi to retrieve this information. This is done through a RESTful web service.

iPhone App Progress

Jordan Doell

I got the framework from James. Now I am beginning to connect the framework to the user interface I had before. Once I get that done, I can start trying to send JSON over to the server that the Raspberry Pi will talk to.

Most of my time spent during this sprint was looking through the framework we got from James. I had to go through and try to understand what all is going on and how to use it. Now I am beginning to connect up the interface. Also, I have been researching about JSON. I haven't used JSON before so I wanted to be a little more familiar with it. As soon as the interface is done, I can start trying to send some JSON data to the server that Austin has been working on.

Meetings

Since the sprint started, we have met a few times after Senior Design class on Tuesdays and Thursdays. Jordan also met with James and Josh when he got the framework and James described it briefly.