Christmas Lights Animation

Requirements and Research Document (Concept of Operations)

Prepared By:
Austin Wentz
Jordan Doell

Revision History

Date	*Author*	*Version*	*Comments*
12/12/12	*Jordan Doell*	*1.0.0*	*Initial version*
4/20/13	*Jordan Doell*	*1.1.0*	*Updated user stories and other sections*
4/23/13	*Jordan Doell*	*1.2.0*	*Added Research Activities, Updated User Stories*
4/23/13	*Jordan Doell*	*1.3.0*	*Added Sprint Report Information*

Table of Contents

1.0 Overview

This document will give the reader a good background of knowledge of the requirements and other various details about the Christmas Lights Animation Project. This document is a living document, meaning that it will be added to and revised throughout this project.

1.1 Scope

This document contains stakeholder information, initial user stories, requirements, proof of concept results, and various research task results.

1.2 Purpose of the System

To make an interactive Christmas lights animation product that synchronizes Christmas lights to music. It will be controllable from an iOS device.

1.3 Project Description

Use an embedded Linux single board computer such as a Raspberry Pi ($35) or BeagleBone ($89) as the Christmas light's controller. Both boards have a complete software development environment on them to write the controller code in C++ or Python. Same hardware as before, with the addition of circuits that are able to switch 120VAC. An 8 channel switch that can be controlled via GPIO pins is available on the internet for about $20-$30. The simple version will merely switch lights on and off according to some predefined sequence. A more advanced version will allow the lights to be synchronized to music. In this case, the smartphone controller is more likely a PC/Mac running a MIDI sequencer. (Google “MIDI”) Each light could be represented as a specific MIDI note. Your Linux board then looks like a MIDI instrument to the sequencer. You use the sequencer software to “program” the light show.

2.0 Stakeholder Information

L-3 is the main sponsor of the project, with June Knight being the contact.

2.1 Customer or End User (Product Owner)

The customer is June Knight of L-3 Communications, who will own all I.P.

2.2 Management or Instructor (Scrum Master)

Dr. McGough will act as Scrum Master for our project. We have sprints in 3 week cycles, with weekly meetings scheduled.

2.3 Developers | Testers

Jordan Doell and Austin Wentz are the developers and testers. Austin has been taking charge of the hardware side, and Jordan is handling the iOS application.

3.0 Requirements and Design Constraints

The XBMC plugin will run on most platforms, including Linux, Mac OSX, and Windows. The iOS application will run on most iPhones, iPads, and iPod Touches. For the iOS app to work, the device must have an internet connection. Also, the server and machine running XBMC will also need an internet connection. The server will need a static IP address as well.

3.1 System Requirements

- Sync Christmas lights to music
- Control with an iPhone app

3.2 Network Requirements

- Connect to the system from any wide area network

3.3 Development Environment Requirements

Development for the iPhone app is done in XCode on a Mac. The development for the hardware is done on a Windows pc. Most platforms should work as long as Python is able to be run on the system.

3.4 Project Management Methodology

The project is managed using the Agile methodology Scrum. The Scrum Master is Dr. Jeff McGough. Sprints are 3 weeks in length and weekly meetings are held. Trello is used for managing the backlog. All parties should have access to the backlog and sprint reports. For this project, there will be 6 sprints. Each sprint is approximately 3 weeks in length. All code and documents will be uploaded to the Github repository.

4.0 User Stories

The following are the user stories for our project. They are the non-technical descriptions that our system should be able to.

4.1 User Story #1

The user would like to be able to sync Christmas lights to music.

4.2 User Story #2

The user would like to control the system on an iPhone.

4.3 User Story #3

The user would like to control the system from anywhere with an internet connection.

5.0 Research or Proof of Concept Results

The following sections show our research activities, and provide our proof of concept results from the project.

5.1 Research

There has been quite a bit of research involved for this project. Jordan has not had any previous experience developing an iPhone app. So, he has gone through several podcasts and examples to learn Objective-C programming and using the XCode IDE. Austin has had to research different SSR's and controllers and how to solder them together. Also, he has had to learn about using the Raspberry Pi board, along with controlling the lights with software.

5.1.1 iOS Application Research

For learning iOS programming, Jordan went through a podcast on iPhone and iPad application development. The podcast was available on iTunes for free in the iTunes U section. The course was titled "iPad and iPhone App Development (Fall 2011)."

Link to podcast: https://itunes.apple.com/us/course/ipad-iphone-app-development/id495052415

Also, multiple online resources and tutorials were used for working with XCode and other general aspects of iOS programming. James Wiegand was also a huge help with the app. He sat down with me and Josh Kinkade and lead us through some example apps. He also made a general networking framework that was initially intended to be used as a guideline framework for our application. However, after a few issues getting things connected and put together, I felt I had gained enough knowledge to design do the app on my own without the need for James' framework.

5.2 Proof of Concept

We have completed the project successfully. All of the Requirements have been met and the system is working. There is a proof of concept, meaning that we can control the system with the iPhone app,

with the lights being sequenced to the music. The commands are being sent to the server from the iPhone. The server is receiving the commands and passing them on to the XBMC plugin which is then able to control the lights. The songs are presequenced beforehand and then played when commanded.

Appendix I: Supporting Material

This document might contain references or supporting material which should be documented and discussed in appendices. This material may have been provided by the stakeholders or it may be material garnered from research tasks.

Appendix II: Sprint Reports

This section will contain a complete list of all of the period progress and/or sprint reports which are deliverables for the phases and versions of the system.

II.1 Sprint 1 Progress Report

Sprint Report 1

Team Members: Austin Wentz and Jordan Doell
Date: October 5, 2012
Class: Senior Design
Subject: Sprint 1 Report
Sponsor: L-3: June Alexander-Knight

Sponsor Description:

L-3 Communications is a world class defense contractor. They play a huge role in the defense industry for the United States government. June Alexander-Knight graduated from SDSMT and since then, works for L-3. She has also been a strong supporter of SDSMT students and graduates.

Sponsor's Problem/Goal:

Sync Christmas lights to music using a Linux board and controller, and control the system using an iPhone app.

Customer Needs:

- Linux board to control lights
- SSR's to power on and off the strands of lights
- iPhone app to do sequences or play music
- Use sequencer software to program light show

Project Environment

Project Boundaries

- The project will have two separate environments: mobile device environment and Christmas lights controller environment
- The project's mobile environment will be focused on iOS devices
- The project's controller environment consists of Raspberry Pi, PIC microcontrollers, and additional circuitry to control the lights

- Communication between environments will be done over TCP/IP via JSON
- The mobile environment will be developed in Objective-C
- The controller environment will be developed in Python, and also in Clojure
- No code will need to be written for the PIC microcontrollers

Project Context

Technical Environment

The technical environment can be split into three parts: mobile device, high-level controller, and low-low-level controller.

Mobile Device

The iPhone is used as the mobile device. Development will be done on a Mac mini.

High-Level Controller

The Raspberry Pi is used as a high-level controller. It will receive commands from the mobile device, perform any required processing on command data, and send the commands to the low-level controller. The Raspberry Pi uses a Debian-like flavor of Linux. Development will be done in Linux and Windows.

Low-Level Controller

To directly control the Christmas lights, we are using a popular do-it-yourself light dimmer scheme called Renard. In particular we are using the Renard 64 XC design. No development needs to be done on the low-level controller.

Current Systems Overview

Figure 1: System Overview

Product Deliverables

No product deliverables at this point.

Future Product Deliverables

- Functional prototype
- Source code
- User manual / documentation
- Requirements document
- Design document

Backlog

Completed

- Purchase and configure single board computer (SBC) to act as high –level controller
- Purchase SSR pcb kit, SSR heat sinks, and Renard microcontroller pcb kit
- Analysis and research for design and requirements for project
- Start learning iOS development

Remaining

- Develop interface between Raspberry Pi and Renard 64XC
- Implement Renard serial protocol
- Develop prototype which switches lights on and off using predefined sequence
- Assemble additional circuitry (SSR and Renard kits)
- Purchase Christmas lights
- Purchase extension cords
- Program and configure Raspberry Pi to act as midi sequencer for lights
- Develop and implement iPhone app which controls the Christmas lights

Potential Issues

- Difficulty in assembling additional circuitry correctly
- Troubleshooting issues with SSRs and Renard microcontroller
- Safety issues when dealing with high voltage power sources
- Possible issues with iOS development

II.2 Sprint 2 Progress Report

Sprint 2 Report

Team Members: Austin Wentz and Jordan Doell
Date: November 1, 2012
Class: Senior Design
Subject: Sprint 2 Report
Sponsor: L-3: June Alexander-Knight

Backlog

Completed

- Purchase and configure single board computer (SBC) to act as high –level controller
- Purchase SSR pcb kit, SSR heat sinks, and Renard microcontroller pcb kit
- Analysis and research for design and requirements for project
- Start learning iOS development
- Assemble additional circuitry (SSR and Renard kits)
- Implement Renard serial protocol
- Purchase Christmas lights
- Purchase extension cords
- Develop prototype which switches lights on and off using predefined sequence
- iPhone app prototype

Remaining

- Design display case for electronic components
- Have the display case made and assembled.
- Program and configure Raspberry Pi to act as midi sequencer for lights
- Develop and implement iPhone app which controls the Christmas lights

iOS Application progress:

Jordan Doell

During Sprint 2, I have been continuing to learn Objective-C and iOS application development. I found and have been watching a podcast that covers iOS development and Objective-C. Also, James has been lecturing to me and Josh about iOS and some of the components we will need for the project. We still have a few more lectures to go, but we are making progress.

App Prototype:

I have gained enough knowledge of iOS so far to make a simple prototype. It is nonfunctional so far, but gives a little direction to where we are headed with the app. Below are some screenshots of the different views in the app.

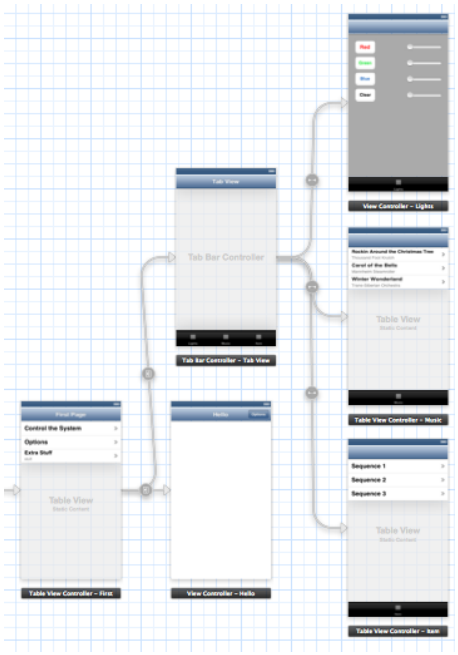


Fig. 1: Overall storyboard for the prototype

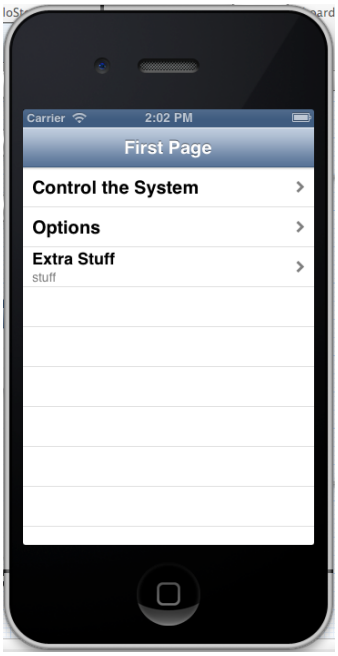


Fig. 2: Main page of the app

Fig. 3,4,5: Lights tab, Music tab, and Sequences tab

Christmas Light Controller Progress

Austin Wentz

Considerable progress has been made on the hardware front. The Renard 64XC and the 8 SSRez's are now soldered and thoroughly tested. In total, the soldering took 20-25 hours. Testing took another 5 hours to complete. With the hardware assembled, I put together a simple prototype which turns lights on and off using a predefined sequence. Several short videos are available to demo the prototype.

Display Case

I have also been working on a design for a display case which houses the hardware. The dimensions of the case will be 16.5 inches x 16.5 inches x 12 inches. Here are some initial requirements for the case:

- Safety features – Renard 64XC and SSRez's will only be powered when lid is closed.
- Locking mechanism to prevent theft
- Made of acrylic
- Fan for keeping SSR's cool
- Cord management

II.3 Sprint 3 Progress Report

Sprint 3 Report

Team Members: Austin Wentz and Jordan Doell
Date: December 7, 2012
Class: Senior Design
Subject: Sprint 3 Report
Sponsor: L-3: June Alexander-Knight

Backlog

Completed

- Put Christmas lights on house for demo
- Film demo of Christmas lights blinking in sync to music
- Design display case for electronic components
- Have the display case made and assembled (went with pre-assembled case)

Remaining

- Program and configure Raspberry Pi to act as midi sequencer for lights
- Develop and implement iPhone app which controls the Christmas lights

Christmas Light Controller Progress

Austin Wentz

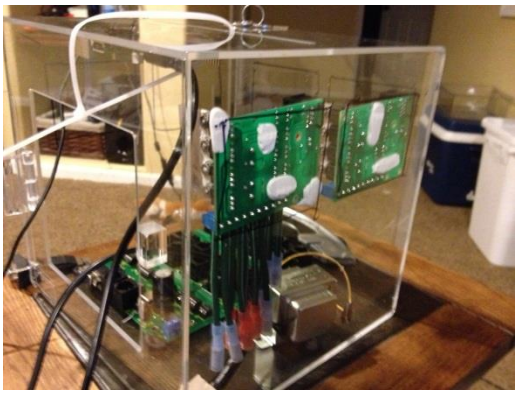
We made substantial progress on the Christmas light controller during Sprint 3. A display case was purchased to house the embedded hardware, I put up Christmas lights on my house, and a demo was filmed of the Christmas lights blinking in sync with music.

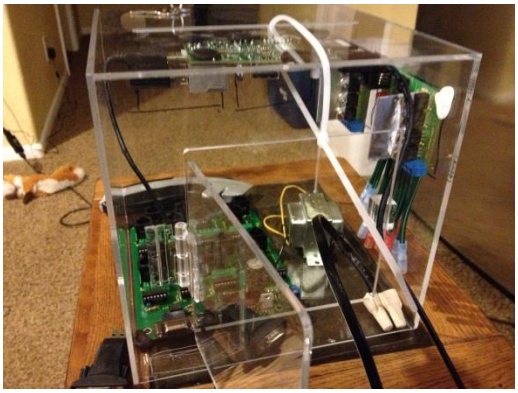
Display Case

Designing a display case and having it assembled was taking more time than originally anticipated, so we went with a premade temporary solution. An 8" x 8" x 8" acrylic display case, originally designed to be a ballot box, was purchased and modified. Before and after photos are shown below:

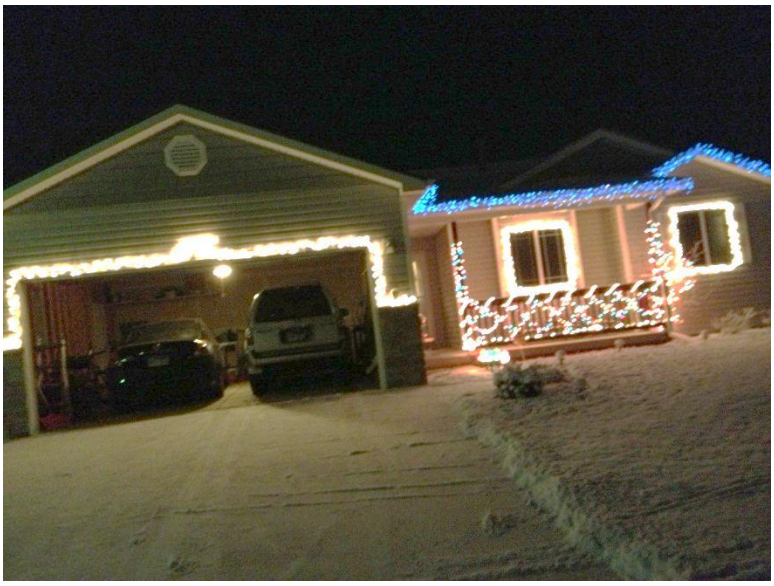
Before

After





Christmas Lights



iPhone App

Jordan Doell

Prototype

The prototype app GUI was primarily made during sprint 2, but some of sprint 3 was spent researching some more of how we want the app to look. Hopefully during Christmas break, some more progress will be made.

Also, more time was spent learning a little more about iOS. I'm still getting through the podcast class, so hopefully the app will begin gaining some functionality soon. James is putting the final touches on his framework, so as soon as he gets that finished, that will be put into the app as well. The communication between the iPhone and base station will probably be the biggest challenge to get working.

II.4 Sprint 4 Progress Report

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 2

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.

II.5 Sprint 5 Progress Report

Sprint 5 Report

Team Members: Austin Wentz and Jordan Doell
Date: March 15, 2013
Class: Senior Design
Subject: Sprint 4 Report
Sponsor: L-3: June Alexander-Knight

Backlog

Completed

- Implement client code on Raspberry Pi/laptop
- Refine RESTful web service on EC2 server
- Send JSON from iPhone to server
- Connect iOS framework to UI

Remaining

- Test web service
- Test client code

Christmas Light Controller Progress

Austin Wentz

EC2 Web Service Implementation

For the web service, we are using Flask which is a microframework for Python web development. Song and light sequencing information is stored in a sqlite database. Retrieval and adding/updating information is done through GET and POST commands. For example, to retrieve a list of available songs to play, just send a GET request at the /songs URL.

Client Application Implementation

The client application is implemented in Python also. The application has several different modes. One mode is to change the brightness of the lights based upon values on the web service. The values can be modified via mobile devices. Another mode is to receive commands to play songs and/or light sequences.

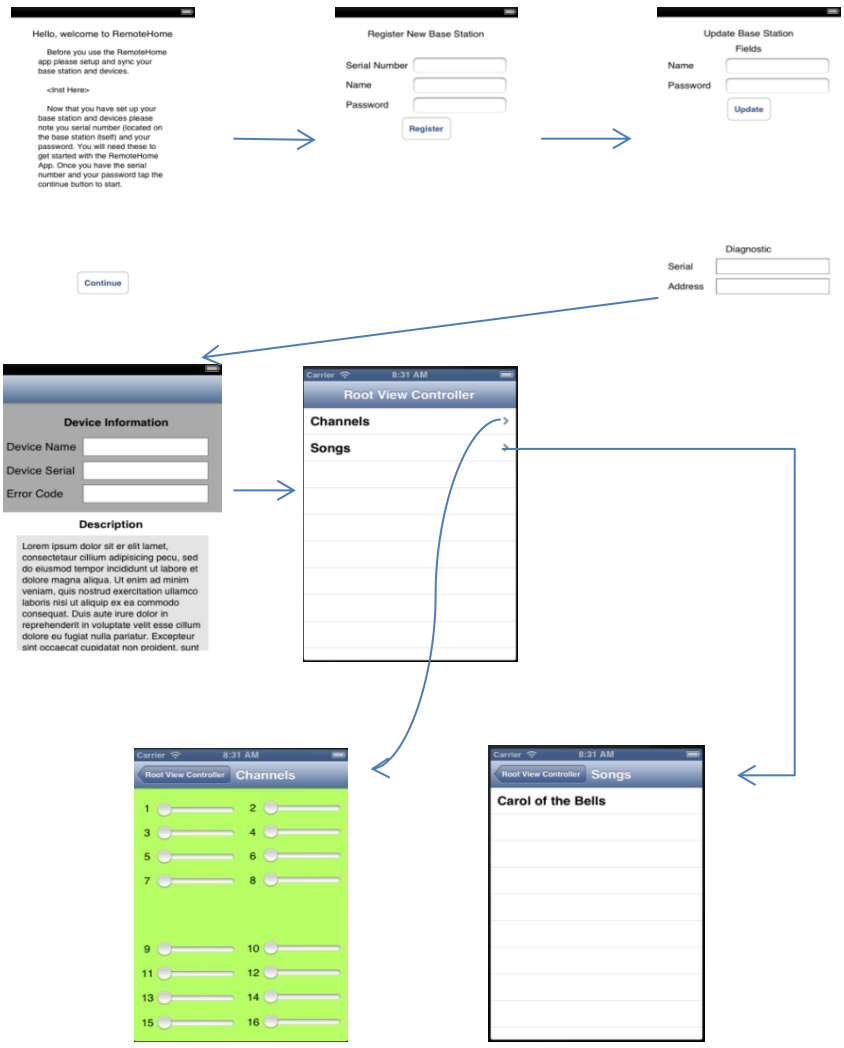
iPhone App Progress

Jordan Doell

For the app, I have included Figure 1 below. I updated the layouts a little bit to simplify things. I am still working on getting the app to flow from James' framework to my interface that I have made. Figure 1 shows what it should eventually look like. All of the first 4 views won't have to be brought up each time the app starts up once the base station gets added initially. The main view with channels and songs is pretty self-explanatory. Once in the channel view, it should send data to the server when a slider has been changed. It will send an array of all the sliders values to the server. For the songs, they will be hard coded for now. When a song is tapped, it should send data to the server letting it know which song to begin playing.

I am planning to meet with James or Josh next week to hopefully get everything figured out since I am a little behind. Most of my time this sprint has gone to researching how to accomplish what I need to do in Xcode. Austin has been working on the JSON data for me to send to the server, so once he gets that figured out and I get the layouts connected, hopefully we can begin some testing.

Figure 1:



II.6 Sprint 6 Progress Report

Sprint 6 Report

Team Members: Austin Wentz and Jordan Doell
Date: April 12, 2013
Class: Senior Design
Subject: Sprint 6 Report
Sponsor: L-3: June Alexander-Knight

Backlog

Completed

- Some Testing
- Presequenced the songs and made midi files with Vixen
- Finished integrating lights into XBMC
- Created display for Design Fair

Remaining

- Testing

Christmas Light Controller Progress

Austin Wentz

During sprint 6 I finished integrating the light sequencing into XBMC. Sequences and their associated songs can now be played within XBMC. Since we are using Vixen to create the sequences, I wrote a utility program that converts the sequence from Vixen's format into our own format used within our plugin for XBMC.

iPhone App Progress

Jordan Doell

I changed up the UI for the app in between sprints 5 and 6. Also, it sends out commands to the server now. This uses the JSON data format. The storyboard is shown in figure 1.

Most of the other time spent during this spring was for some testing and preparation for the design fair. This included figuring out what to do for the poster and designing it. The poster is almost complete. There are a few changes that need to be made but that shouldn't take too long. Also, we are still figuring out how we want the design to look at the fair, like where the lights should go and how to organize our booth.

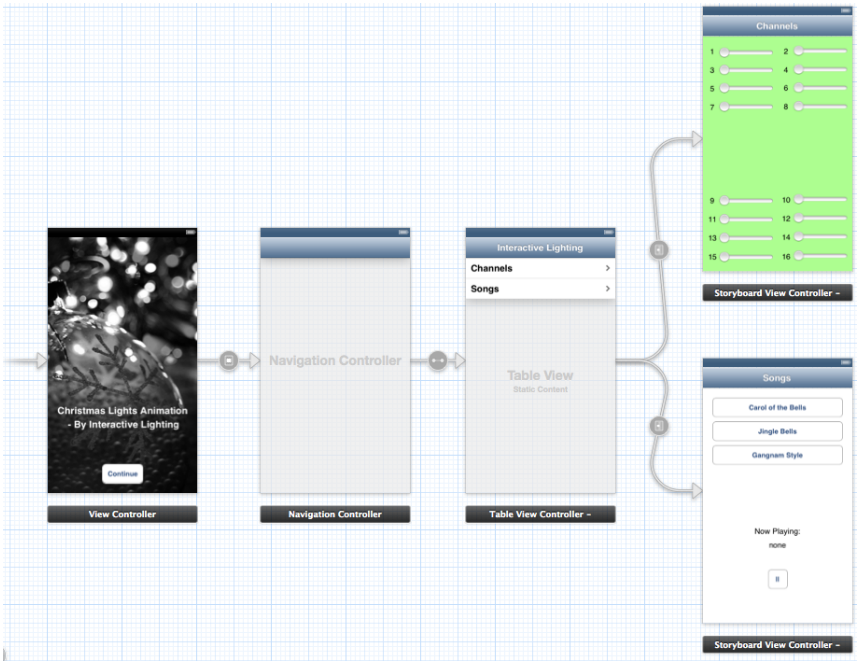

Channels
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
Storyboard View Controller –
Christmas Lights Animation
- By Interactive Lighting
Continue
View Controller
Navigation Controller
Navigation Controller
Interactive Lighting
Channels
Songs
Table View
Static Content
Table View Controller –
Songs
Carol of the Bells
Jingle Bells
Gangnam Style
Now Playing:
none
II
Storyboard View Controller –

Figure 1: