Cameron Currey
Sam Nazari
Tim Pruss

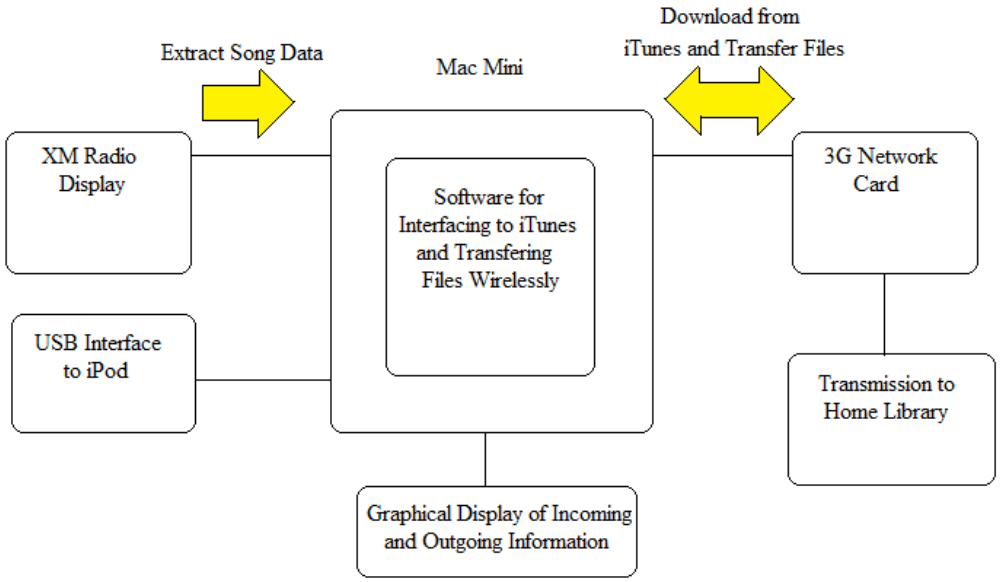
The iCarTunes Project Proposal

Project Introduction

Our project will be designed and built to allow users to wirelessly download songs that are currently playing on the radio from the popular iTunes network in an automobile. Once the song is downloaded the user will have the option to load the song onto an iPod via a USB cable or to wirelessly transmit the song and store it on a home library.

Project Components

The project is designed to have several components. The components will include an XM Radio Interface, 3G Wireless Network Card, Mac-Mini central processing unit, USB interface to iPod, and a graphical display for the user.



Description of Each Component

Mac-Mini

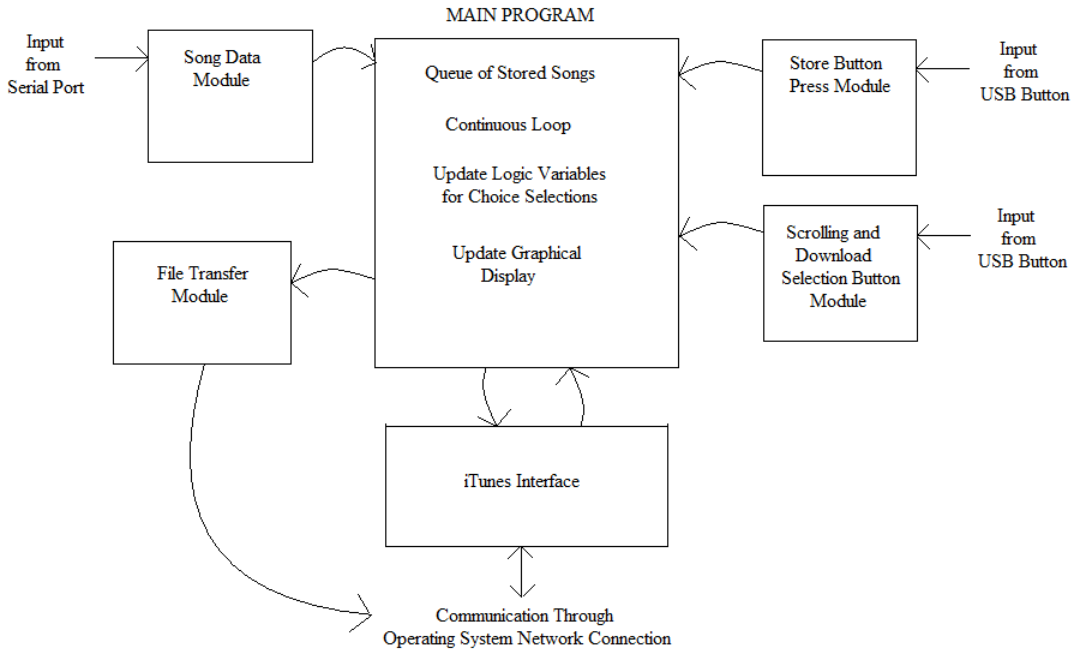
The main processing unit of the entire system will be using the *Mac-Mini*. The Mac-mini is a small computer that will fit in a car and provide an easy interface to all the components to the system. The Mac-Mini is also very easily programmable and will be capable of executing C# code. The Mac-Mini has an operating system of itself so it will have capabilities to communicate to the Internet through a 3G Network Card as well as run the iTunes software. The Mac-Mini is capable of running the iTunes software, so the developers of this project will not have to reverse engineer the proprietary iTunes software.

The software that runs on the Mac-Mini will be responsible for the following:

1) When a user button is pressed, the data about the current playing song will be stored in a queue of preferred songs.
2) Send the song information to the iTunes search query and display the results
3) Allow user to download a desired search result selection.
4) Allow user to make selection to transfer song wirelessly to home library
5) Allow user to make an option of transferring song to connected iPod

All of these functions of the software will be implemented using C# programming language. The software will be developed using Microsoft Visual Studio licensed version at the University of Utah. This software will be the core of the project and will need to interface with the secondary components.

The implementation of the software will be refined during the development of the project but the following diagram will model the core of the software:



Main Program

The main program's function will be a continuous loop to update and display all incoming information from the various modules. For example, if the song data module updates the queue of stored songs, the main programs function will be to display those in the graphical interface. If a button is pressed, then the next iteration over the loop, the song data module results will be loaded into the queue. The overall main function of the main program is to logically handle input from the various modules.

Song Data Module

The Song Data Module's will be responsible for deciphering the input from the Serial Port that is connected to the XM radio and converting it into a format easily usable for the graphical display and easily usable by iTunes. The input from the Song Data Module into the main program will be a string containing the song title and artist name.

Button Press Modules

These modules will be relatively simple. They will update variables in use by the main program signifying whether a button has been pressed or not. Once these variables have been updated, the main program will be able to make a choice about which action to take.

iTunes Interface

The iTunes interface will be a scripted language for the software to interface with iTunes software running in the operating system's background. The iTunes interface will use the scripting AutoIt that will manually perform the requested operations in the iTunes software. For example, when a search for a specific song is made, an AutoIt script will be executed that will paste the song information into the iTunes search query then extract the search results and return it to the main program.

AutoIt is a freeware-licensed application that is available for download on the Internet.

Graphical User Interface

The GUI for the software will be very intuitive. The main menu of the GUI will allow for the user to view saved songs or option to transmit songs or load onto a connected iPod. Within the saved songs sub menu it will simply display the list of songs and allow the user to scroll up and down through the choices.

The Transmit Song menu will allow the user to make an option between transmitting to an iPod or to a home computer. Once a transmit option has been selected, the user will then be able to scroll through the list of songs that have been downloaded and make a selection of which to transmit.

Mac-Mini Specifications and Advantages versus a Microcontroller

The Mac-Mini weighs 2.9 pounds which will make it ideal for portable use in a car. It is 2.5 x 6 inches so it should fit conveniently under the dash of any automobile.

The Mac-Mini also has an 80GB hard drive which will support plenty of songs ready for download.

The Mac-Mini has 4 USB ports which will be used for a button input device, the Wireless Card connection, and the input from the Serial Port to USB connection device to the XM radio display.

The Mac-Mini also has a battery life of one hour so it can be used even without the voltage from the automobile connectors.

XM Radio Display

An XM Radio display displays the song title and artist of the song currently playing. This information from the display screen in the car will need to be extracted and piped into the Mac-Mini via a serial cable. Once this information is available, it will be usable by the software to determine what to do with the song.

We will use a Delphi SA10276-11P1 XM Radio display to extract this information. After we acquire the radio display, we will disassemble the radio and add wire taps to the ASCII converter for the LED display. The ASCII converter is a mechanism in the XM radio that converts data into a format to be displayed on the LED. Wire taps will be placed on the input of the ASCII converter and connected to a serial port cable. In the software portion of the project, the Song Data Module will collect this information and make it a format usable by code.

Schematics for the Delphi SA10276-11P1 can be found online for us model the extraction.

3G Wireless Card

The 3G Wireless cards provide satellite Internet communication to Internet provider networks. Once we have working software and hardware interface we will need to activate an account to provide satellite access.

The network communication will be responsible for downloading from iTunes and our software running on the Mini-Mac will need to send files through the Internet software running on a home computer. Due to the operating system running on the Mini-Mac setting up this network should be relatively easy.

We will use an AT&T USBConnect 881 3G Wireless network card. The card has a USB connection that will allow us to set up an Internet connection on the Mac-Mini. This will eliminate many of the hardware interfacing issues we had with the initial concept of interfacing with the 3G Wireless Card.

The specifications of the card are as follows:

- Weight: 1.25 ounces (with battery)
- Dimensions: 3.7 x 1.5 x 0.8 inches
- Connects to standard USB port
- Tri-band HSPA - 850/1900/2100 MHz
- Quad-band GSM/GPRS/EDGE - 850/900/1800/1900 MHz
- EDGE Class 12 - download speeds of up to 135 Kbps
- Built-in antenna improves 3G performance
- Includes integrated status indicator lights
- OS support for Microsoft Windows Vista, XP, and 2000
- Compatible with Mac notebooks (Mac OS 10.4 or later)
- Get a free Mac client at: http://www.sierrawireless.com/support/att/mac
- Warranty: One year on new devices, 90 days on refurbished

Once the hardware is setup for the system, we will need to setup a wireless account through AT&T 3G Wireless Network.

Graphics Display

To provide functionality for our entire system we will need a mini-display screen to display all these results. We can connect the Mac-Mini to an alternate display device to facilitate testing, but if the budget allows we would like to incorporate a Macintosh Mini-DVI 128x128 Pixel display screen. This display would provide the functionality needed to display the results of our software and allow the user to make selections about results and connects easily to the Mac-Mini.

Power Converter

The Mac-Mini has an expected battery life of approximately one hour. If time permitting we would like to incorporate a power converter to run the Mac-Mini from a car battery to provide extra mobility of the iCarTunes project.

To build such a converter we will need an OPUS connector. OPUS is a device that allows connectors to a standard car battery to run an AC device. Attaching an OPUS to a cigarette lighter or car battery will allow for a reliable power source for the Mac-Mini.

Software Running on Home Computer

The final component of the project involves software simply running in the background of a home computer connected to the Internet. This software's purpose will be to listen on a specific port and receive all incoming songs being sent to it from a system sending such a transmission and store them in a designated folder. This will allow these songs to be stored and readily available to use from your home computer.

The software to run on a home computer will be developed jointly with the transmission software to be run on the Mac-Mini. It will be developed using C# with Microsoft Visual Studio. The software will interface with a specific port and use a byte transfer protocol. When each byte is received the will be written into a file. Once the file is complete transferred and written it can be stored in the iTunes directory.

Car-Bundle Package

When the project is fully implemented and all components are functioning properly we will encase all components in a plastic case to allow a complete system to be placed into a car. The package should fit within the confines of a radio display and be easily usable.

Predicted Budget

Mac-Mini	$600
AT&T USBConnect 881 3G Wireless Card	$33
Delphi SA10276-11P1 XM Radio Display	$30
Additional Adapters	$20
Satellite Connection per month	$30
Total	*$713*

The overall cost of our entire project seems to be a bit much for 3 people in a single project. It is our intention to write letters of request to various retailers for any possible donations to our project. We also know a person who may be able to provide us support with a Satellite Connection at AT&T.

We will write a proposal to the Apple Store for possible donations or rentals of a Mac-Mini we could use to build the project.

Schedule Flow

The following is a tentative schedule of the project and the desired flow of development of the project. The ordering of these components must be done

May 5 – Write letters of request to retailers for donations to project
May 12 – Purchase XM radio display and research its schematics
May 19 – Purchase/pickup Mac-Mini and connect serial port from XM radio display
May 26 – Interface software to extract data when a button is pressed
June 2 – Write skeleton software and test ability to extract song data
June 9 – Modify software to hand off Song data to an iTunes search query
June 16 – Modify Software to display search results and modify button Interface
June 23 – Test song download capabilities
June 30 – Build customized GUI to be displayed on display screen
July 7 – Develop software to run on home computer to receive file transmissions
July 14 – Purchase display screen and test Mac-Mini capabilities
July 21 – Purchase and set-up 3G Wireless Internet Connection on Mini-Mac
July 28 – Test system and debug any issues
August 4 – Testing
August 11 – Bundle system in a car-ready package

Project Risks

The biggest risk of our project is the estimated price. Each component is fairly expensive. However, with the choice of using a Mac-Mini we can actually begin development of any of these components on any other machine and port it to

the Mac-Mini later without making any purchases. The XM Radio display will be the first piece of hardware we will need, and fortunately it is relatively inexpensive so we can begin development on that right away. In fact, our entire project could be developed on any .NET capable machine, and later be ported to a Mac-Mini.

It is our hope we can convince various businesses or retailers to donate any hardware to our project by presenting our project and requesting for any aid they may provide to our project at the University of Utah.

Another risk is our lack of knowledge of XM radio displays. Nobody in the group is aware of how we can take apart a radio display and pipe the information into a serial port. This will require some research and careful thinking before we are able to proceed to the next part of the project. Again however, this information could easily be simulated for the development of the further advancement of the project.

The choice of using a Mac-Mini as the main CPU was a good choice. It has resolved many of the potential interfacing issues, and has made each component of the project independent of the other. So if development on one aspect of the project is held up, progress can still resume in other components.