International Journal of
Emerging Research in Management &Technology
ISSN: 2278-9359 (Volume-4, Issue-12)

Research Article | **December 2015**

Methodology to Design Network Music Player iPhone Application

Gypsa Agarwal
Information Technology & SITM,
Lucknow (UPTU) Uttar Pradesh,
India

Ravendra Ratan Singh Jandail
Dept. of Computer Science/IT & SITM,
Lucknow (UPTU) Uttar Pradesh,
India

Abstract—

iPhone is the mobile device, a product of Apple Inc. with iOS as its operation system. Apple provides tools, IDE and frameworks for third party apps developer to develop, distribute and deploy their apps on app store. In this paper we propose the methodology to develop a network music player app for iPhone/iPod. The application objective is to play an audio in more than one device simultaneously in a synchronised manner which can be controlled and configured from a single device. The application is based on the client server architecture. The paper describes the methodology, tools used to develop the application.

Keywords*— *iPhone App, Objective-C, Bonjour, Client-Server

I. INTRODUCTION

There are lot of applications developed and deployed on app store for iPhones/iPod's and iPads. The application is developed in apple native technology using XCode and Objective C. Application concept is to develop a music player application using Bonjour. Bonjour is a technology that makes the discovery of services very easy. Bonjour works very well with the CocoaAsyncSocket [8] library, an open source library that provides an Objective C interface for working with sockets on iOS and OS X [7].

As described by Apple, Bonjour is Apple's implementation of Zero-configuration networking (Zeroconf), a group of technologies that includes service discovery, address assignment, and hostname resolution [1]. Bonjour locates devices such as printers, other computers, and the services that those devices offer on a local network using multicast Domain Name System (mDNS) service records [1]. Bonjour is in charge of publishing and discovering services on the network. Bonjour is not responsible for establishing a connection between the server and the client [1].

The application uses Client-server model, in the client server model, resources are provided by the server and the client requests the resources from the server. A server host runs one or more server programs, which share their resources with their client. The clients initiate the communication session with server, which waits for incoming request. Client and server exchange messages in a request-response messaging pattern: The client sends a request and server returns a response. A server may receive request from many clients simultaneously or in short period time, computer can perform limited number of tasks at any moment, and scheduling system is incorporated to accommodate them in turn.

For network sync, the communication is established through Sockets."Socket is used to send and receive data". A socket is one end of a communication channel between two processes that want to talk to each other. A network connection (or interprocess communication channel) has two sockets, one for each end of the channel. A network connection is established by one socket making a connection with another socket, the listening socket, which is listening for incoming connections [5].

A socket address is the combination of the IP address and a port number. Based on this address, the internet socket delivers incoming data packets to the particular application process/thread.

The proposed application can be installed on the iPhones/iPods. The users can install the application in a group and can play song with effect of home theatre system and surround setup without any other special equipment.

II. PROPOSED CONCEPT

The application is based on the client server architecture where one device is a server and other devices are clients. After installing the application user can choose to either be a server or a client. The server broadcast himself in the network and the clients can see the list of servers in the network, the client can join any one of the available networks. The server can see all the client devices and the client can see all the server devices. The service discovery mechanism is done through Bonjour. Bonjour offers discovery of devices connected to the same network. After the devices are discovered, the connection is made between the server device and the client device through TCP protocol. Once the connection is established between all clients and server, can server scans the music library and select the song for playback, the server sends the song on all the clients devices, the song is saved on all the client devices, and the server send the READY and then the PLAY command to all clients and the playback is started on all client devices. The song is played using apple AVFoundation framework. After the song/playlist playback finished, the connection is aborted and the media is deleted from application. The media is deleted to save the memory space for now. To ensure the synchronous playback, we have

incorporated a method to find the ping time in all clients and from all round trip time, calculated the highest mean delay value and then scaled the highest mean delay value with particular client delay and find the correct delay value for every client so that, client when receives the play command with the delay value, it waits for that delay and play the song after that delay to ensure the server and the other devices are also playing the song at the same time frame [9]. Application also has the refresh feature in case after this method also, there is some second delay exists in playback, the server can refresh and all the devices will be synced again. The server can also share the playlist on all client devices. To give the effect of amplifier or multi channel speakers, the server can set the speaker position.

III. OUR METHODOLOGY

The application to be divided in two modules. There are two major modules of the application:

Host (Server) Implementation

Join (Client) Implementation

A. Host (Server) Implementation

Host module refers to the server implementation in the application. The server module includes broadcasting himself in the network so that the client can view him and can connect him. Whenever the application launches, the user chooses either to host a network or join a network hosted by another device. When the user chooses to host, the application publishes the services using bonjour and CocoaAsyncSocketlibrary. CocoaAsyncSocket library provided by Robbie Hanson (2014) provides easy-to-use and powerful asynchronous socket libraries for Mac and iOS [8].
The major features of the host are mentioned below:

Table 1: Host Module Features

Publish a service
Scanning music library
Send music file to all connected clients
Play Song on all connected clients
Calibration (refresh feature for synchronous playback)

Publishing a Service

The application automatically publishes a service on the host button action event that other instances of the application on the network can resolve [7]. The class that we will be using for this purpose is NSNetService [4]. It is an instance of the class represents a network service. The API of NSNetService provides a convenient way to publish the services offered by the application and to resolve the socket address for a service [4]. To publish a service, a port is acquired and a socket is prepared to communicate with clients. 0 is passed as a port number so that is upto O.S. to supply the port. Service is published via initializing NSNetService object with the service name, domain, type, and port information [4] calling method - initWithDomain:type:name:port:. Once initialized, publish method is called to broadcast the service information to the network. NSNetServiceDelegate Protocol is conformed to respond messages and handling error.

Sending Packets to Clients

For sending data from one socket to another socket, the HTTP protocol, which is built on top of the TCP protocol is implemented, The application uses HTTP protocol as descibed by Bart Jacobs(2014) which sends an HTTP header with every request and response. The HTTP header contains information about the request or response, this information is used by the receiver to make sense of the incoming stream of data [6]. HTTP header is the length of the body. If the receiver knows the length of the body of the request or response, it can extract the body from the incoming stream of data [6].
In our application, every packet of data that we send through the connection is prefixed with a header that has a fixed length [6]. The header that here contains one piece of information, the length of the body or packet that comes after the header [6]. We have created a Packet class NSMSPacket as refered by Bart Jacobs(2014), which conform to the NSCoding protocol so that it can be encoded and decoded. The class has three properties, type, action, and data. The type property is used to identify the purpose of the packet while the action property contains the intention of the packet. The data property is used to store the actual contents or load of the packet [7].

Music Library Implementation

The application scans the phones music library and displays all the songs available in phone music app. The scanning feature is done through MPMediaPickerController class.
An MPMediaPickerController object, or media item picker, is a specialized view controller that provides a graphical interface for selecting media items. It allows the application to fetch the song list into the app and once the song is selected, the user can save it in its own app. In the application the save song is added in My Tracks. My track is kind of local app library. Server chooses any song and then sends that song to all the clients, a music player view appears on the entire phone screen when the song playback starts. The client device does not privilege to control the player but the server (host) device can stop, play, and pause the song. These functionalities are implemented in MusicPlayerViewController class.

B. Join (Client) Implementation

Join application module covers the other major part of the application, the join module refers to the client implementation in the application. The client module includes the browsing services which are plublished by host. Whenever the application launches, the user chooses either to host a network or join a network hosted by another device. When the user chooses to join, the application searches for hosts and establishes the connection between them.
The major features of the join are mentioned below:

Table 2: Join Module Features

Browsing services
Making Connection
Receive the music file
Music Playback

Browsing services

The application automatically starts browsing services on the join button action event. The class that is using for this purpose is NSNetServiceBrowser [4]. It searches the network for network services.- searchForServicesOfType:inDomain: method of NSNetServiceBrowser is implemented for browsing services. The delegate methods of NSNetServiceBrowser class are notified when a service is discovered.

- netServiceBrowser:didFindService:moreComing:
- netServiceBrowser:didRemoveService:moreComing:

Making Connection

When connecting to a service, NSNetServiceBrowser class locates the service on the network and obtains the corresponding NSNetService object. resolveWithTimeout: method is called to verify that the service is available and ready for our application. If so, the addresses property provides the socket information, which is used to connect to the service.
The below delegate methods of NSNetService provide the callbacks whether the service is resolved successful or not.

- netServiceWillResolve:
- netService:didNotResolve:

A connection is established using GCDAsyncSocket socket creation and connecting to service address. After the connection is made between host and join, the host selects the music file and sends to all the clients. The client receives the music file and store in “My Tracks”. Now the client waits for ready and play messages, when the host sends the play message the song automatically starts on client device. The client does not have privileges to control the music play. It just response over the commands sent by host.

IV. CONCLUSIONS

The result of the research is to further process and proceed the development of iPhone/iPod application. The application can be deploy on app store for users. The methodology and the proposed concept is used to develop the iPhone application.

ACKNOWLEDGMENT

It is a pleasure to acknowledge my gratitude and thanks to my supervisor Ravendra Ratan Singh Jandail, who provided the motivational guidance during the entire preparation of this project, answering a number of our technical queries despite his busy schedule. His valuable suggestions, constructive criticism and timely help proved extremely fruitful.

REFERENCES

[1] Apple Inc., Bonjour Overview, [Online] Available at:https://developer.apple.com/library/ios/documentation/Cocoa/Conceptual/NetServices/NetServices.pdf
[2] Apple Inc., Discovering and Advertising Network Services, [Online] Available https://developer.apple.com/library/ios/documentation/NetworkingInternetWeb/Conceptual/NetworkingOverview/Discovering,Browsing,AndAdvertisingNetworkServices/Discovering,Browsing,AndAdvertisingNetworkServices.html
[3] Apple Inc., iOS Technology Overview, [Online] Available at: https://developer.apple.com/library/ios/documentation/Miscellaneous/Conceptual/iPhoneOSTechOverview/iOSTechOverview.pdf
[4] Apple Inc., NSNetServices and CFNetServices, [Online] Available at: https://developer.apple.com/library/mac/documentation/Networking/Conceptual/NSNetServiceProgGuide/Introduction.html
[5] Bart Jacobs, Creating a Game with Bonjour - Networking Overview, [Online] Available at: http://code.tutsplus.com/tutorials/creating-a-networked-game-with-bonjour-part-1--mobile-16163
[6] Bart Jacobs, Creating a Game with Bonjour - Sending Data, [Online] Available

at:http://code.tutsplus.com/tutorials/creating-a-game-with-bonjour-sending-data--mobile-16437
[7] Bart Jacobs, Creating a Game with Bonjour –Client and Server Setup, [Online] Available at:http://code.tutsplus.com/tutorials/creating-a-game-with-bonjour-client-and-server-setup--mobile-16233
[8] Robbie Hanson, CocoaAsyncSocket, [Online] Available at: https://github.com/robbiehanson/CocoaAsyncSocket
[9] Valve Developer Community, Source Multiplayer Networking, [Online] Available at: https://developer.valvesoftware.com/wiki/Source_Multiplayer_Networking

