**Pseudo code**

**General**

**Initialize Program** Create GUI elements  
 Start Winsock  
 Disable Server Specific Elements  
 Load last used playlist

**Wait for user Input** if Server button pressed  
 Go to Server  
 if Media button pressed  
 Go to Playback  
 if Client button pressed  
 Go to Client  
 if Exit button pressed  
 WSACleanup  
 Exit Program

**Server***See Server Pseudo code*

**Client***See Client Pseudo code*

**Playback***See Playback Pseudo code*

**Server**

**Wait for User Input** Broadcast button pressed  
 if (broadcasting)   
 unset broadcasting flag  
 else   
 Start Thread with function: Connect to Multicast group

**Connect to Multicast group**  
 Create UDP socket  
 Set broadcast option  
 Fill in Address structure for broadcast  
 Go to Check for enough data to send

**Check for enough data to send** if (file open)  
 if end of file  
 close file  
 go to Broadcast data  
 else   
 go to Check if broadcasting

**Check if broadcasting** if (broadcasting)  
 go to Check for enough data to send  
 else  
 Terminate Thread

**Broadcast data** send udp packet(s) out  
 go to Check if Broadcasting

**Listen Socket** Create a TCP Socket  
 Bind address  
 Set socket to Listen mode  
 go to Accept Socket

**Accept Socket** while servermode  
 perform asynchronous accept call  
 Start new thread with function Wait for Request

**Wait for request** if client requesting list  
 go to Prepare File list for Sending  
 if client requesting file  
 go to Transfer File

**Prepare file list for sending** search for compatible files in current directory  
 add file names to list  
 go to Send List

**Send List** Send list of file names   
 Close Client Socket, Terminate Thread

**Transfer File** Open requested file  
Send requested file  
 Close file  
 Close Client Socket, Terminate Thread

**Client**

**Wait for User Input** if exit button pressed  
 End client mode, Exit application  
 if Listen to broadcast  
 start thread with Connect to multicast channel  
 if Initiate microphone conversation and file related buttons pressed  
 go to Connect to Server  
 if Play music button pressed  
 go to Play Music

**Connect to Multicast Channel** if failed to connect  
 go to Wait for User Input  
 else   
 start thread with function Play Audio from Buffer  
 go to Receive Data

**Receive Data** Block waiting for packet  
 go to Process Data

**Process sound Data** Add data to circular buffer  
 go to Receive Data

**Play Audio from buffer** While playing flag set  
 Check for data in playback circular buffer  
 if data available, play data

**Connect to Server** Get server information from UI Make TCP Connection to server  
 go to Send control message

**Connect to Peer**

Get peer information from UI Make TCP Connection to peer  
 go to Send control message

**Send control message** If (user requesting list of files)   
 send L type control message  
 go to Wait for List  
 if (user requesting file transfer)  
 send filename  
 go to Wait for data  
 if (user requesting microphone chat)  
 go to Wait for confirmation message

**Wait for confirmation message** Block waiting for response from server  
 if (confirmation received)   
 go to Create UDP Socket  
 else go to Wait for User Input

**Create UDP socket** Fill in peer information structure from UI  
 Create the socket  
 if successful, start thread with function Play Audio From Buffer

**Receive Mic Data** Check for data on the socket  
 When data is received, call a completion routine to store the data in the playback circular buffer

**Capture mic data** If data is available from microphone  
 Add that data to the sending circular buffer

**Send Mic Data** Check for data in sending circular buffer  
 If data is available, send data on the socket

**Play Music** Get filename from item selected in UI  
 Play that item (API call)

**Wait for list** Read from the socket  
 If an item is returned  
 Add that item to the media list  
 Else  
 Go to Display media list

**Display media list** Add each item in the media list to the corresponding window

Go to user input

**Wait for Data** wait for file data on socket  
 go to Process File Data  
 if (stream closed)  
 terminate thread

**Process File Data** write data to file  
 go to wait for data