Comm audio pseudocode

# Server

## Initalize GUI

Set up GUI using QT Framework  
Go to **Create TCP socket**

## Create TCP socket

Create TCP socket

Bind address to socket

Go to **Listen for Connections**

## Listen for connections

While true:

If accepted connection

Go to **Update list of clients**

Go to **Create connection thread**

## Update list of clients

Add/remove client name and IP to list of connected clients

If new client

Go to **Create connection thread**

## Create connection thread

Create client connection request thread

Go to **Create UDP socket**

## Create UDP Socket

Create UDP socket

Bind address to socket

Go to **Add to multicast group**

## Add to multicast group

Add client to multicast group

Go to **Send song list**

## Send song list

For each song in multicast group:

Send title

Go to **Wait for instructions from client**

## Wait for instructions from client

While true:

If got "song" type request:

Go to **Song transmission thread**

If got "voice" type request:

Go to **Create VOIP thread**

## Song transmission thread

Create thread

Go to W**ait for song request**

## Wait for song request

Wait for instruction from client

If there is an instruction (song request)

Go to **Send Song**

## Send song

Open song file

While not end of file:

Create datagram

Add song data to datagram

Send datagram to multicast group

Go to **Wait for Song Request**

## VOIP thread

Create thread

Create UDP socket (dedicated to voice)

Go to **Wait for Data**

## Wait for voice data

While true:

If there is voice data

Go to **Send Voice Data**

## Send voice data

Create datagram

Add voice data to datagram

Send datagram to multicast group

# Client

## Initialize GUI

Set up GUI using QT Framework

Go to **Initialize audio output device**

## Initialize audio output device

Search for device

Open device

Go to **Wait for IP & Host Info**

## Wait for IP & host info

If valid IP and host entered

Go to **Create TCP Socket**

## Create TCP socket

Create socket

Bind address to socket

If connect() succeeds

Go to **Connected**

## Connected

Go to **Create data receiving thread**

Go to **Update song list**

## Create song receiving thread

Create thread

Go to **Create ring buffer**

## Create ring buffer

Initialize ring buffer  
Pass ring buffer ID to **Create playing thread**Go to **Wait for incoming server data**

## Wait for incoming server data

While true:

If received server data

Go to A**dd to ring buffer**

## add to ring buffer

Push data to ring buffer head  
Increment head index

Go to **Wait for incoming server data**

## create Playing thread

Create thread

Go to **Check ring buffer**

## Check ring buffer

If ring buffer has data

Go to **Take from ring buffer**

## Take from ring buffer

Pop data off ring buffer tail

Increment tail index  
Play audio

Go to **Check buffer**

## Update song list

Receive song list data

For each song in song list

Update GUI to display song

Go to **Wait for user input**

## Wait for user input

If song requested

Create datagram

Add song identifier to datagram

Send datagram

If “voice” toggled

Go to **Enable voice mode**

## Enable voice mode

Go to **Create voice sending thread**Go to **Create voice receiving thread**

## Create voice sending thread

Create thread

Go to **Initialize mic**

## Initialize mic

Search for device

Open device

Go to **Wait for voice input**

## Wait for voice input

While true:

If got voice input

Go to **Send voice data**

## Send voice data

Create datagram

Add voice data to datagram

Send datagram

Go to **Wait for voice input**

## Create voice receiving thread

Create thread

<Similar to **Song receiving thread**, see above>