|  |  |
| --- | --- |
|  | **2014** |
|  | BCIT  Ramzi Chennaffi  Shane Spoor  Abhishek Bhardwaj |

|  |
| --- |
| **[Comm Audio – Progress Report 1]** |
|  |

# Complete Tasks and Progress Summary

While we did not follow the schedule exactly, we still achieved a large portion of the program’s basic functionality this week. We are confident that we will still complete the application by the deadline. However, should we fall significantly behind, we have the option of removing bonus features such as the text chat and database components to afford extra time.

This week we completed:

* **Server Unicast Send (March 24th, Shane Spoor)**: The server reads a file and sends it in 1KB chunks to a connected client. The file is currently hard-coded; this will be changed next week in implementing the unicast receive function.
* **Client GUI (March 15th, Abhishek Bhardwaj)**: The GUI displays the current playlist, current song and song metadata (artist, album, etc.), a chat window (including audio chat), server options, and the option to save to a file. Depending on time constraints, more features may be added in the future.
* **Audio Playback (March 25th, Ramzi Chennafi)**: The client receives WAV audio and plays it back as it streams. We also plan to add support for other formats (MP3, OGG, etc.) using the different SDL audio API calls.

# Upcoming Tasks

We plan to do the majority of the work in the next week. Since we have a lighter workload in most classes as we near the final exam week, this should be feasible.

The tasks we intend to complete this week are:

* **Client Network Thread (By March 27th, Abhishek Bhardwaj)**: The client will create a network thread to handle the receipt and sending of requests and other data. The thread will also update the GUI to reflect network events.
* **Server Unicast Receive (By March 28th, Shane Spoor)**: The server will receive file requests and voice chat data. On receiving a file request, it will open a file and begin sending to the specified client. Voice data will be sent to any other participants in the voice chat room.
* **Client Receive Audio (By March 29th, Ramzi Chennafi)**: In addition to playing back audio as it streams, the client will also update the GUI with song metadata and a slider showing the song’s progress.
* **Client Request File (By March 30th, Ramzi Chennafi)**: The client will be able to request a specific file from the server. The server will handle this in **Server Unicast Receive** as outline above.
* **Server Connection Manager (By April 1st, Shane Spoor)**: The server will receive requests for audio files and voice data. It will route these requests to the correct completion routines and add them to an internal list detailing the service each client is receiving.
* **Server Multicast Send (By April 1st, Abhishek Bhardwaj)**: The server will send an audio stream to clients who have joined the correct multicast group.
* **Client Save File (By April 2nd, Abhishek Bhardwaj)**: The client will be able to save a file to their local machine as it streams to them.

# Task Summary

The following table summarises the complete tasks, tasks remaining, the people responsible for completion of these tasks, and task completion dates. The stage of completion is indicated in parentheses beside the name.

|  |  |  |  |
| --- | --- | --- | --- |
| Task Name | Task Description | Developer Responsible | Completion Date |
| Client GUI | Features playlist bar, pause/resume button, chat window, and file options | Abhishek Bhardwaj | March 15th, 2014 **(complete)** |
| Audio playback | Client receives and plays back audio | Ramzi Chenaffi | March 24th, 2014 **(complete)** |
| Server Unicast Send | Server sends file chunks successively with completion routine to single connected client | Shane Spoor | March 25th, 2014  **(complete)** |
| Client Network Thread | Client creates a thread to handle sending and receiving | Abhishek Bhardwaj | March 27th, 2014 **(in progress)** |
| Server Unicast Receive | Server receives client command and processes it | Shane Spoor | March 29th, 2014 **(in progress)** |
| Client Send Request | Client can request a file | Ramzi Chennafi | March 29th, 2014 **(in progress)** |
| Client Receive Audio | Client receives and buffers audio | Ramzi Chennafi | March 30th, 2014 **(in progress)** |
| Server Connection Manager | Server receives requests for microphone chat and unicast audio streams and routes them to the correct threads | Shane Spoor | April 1st, 2014 **(in progress)** |
| Server Multicast Send | Server sends an audio stream to multiple clients listening on multicast | Abhishek Bhardwaj | April 1st, 2014 **(in progress)** |
| Client Save File | Client can save incoming audio to a local file | Abhishek Bhardwaj | April 2nd, 2014, **(in progress)** |
| Client Microphone Support | Clients can retrieve microphone data and send it to the server | Ramzi Chennafi | April 3rd, 2014 **(not started)** |
| Server Microphone Support | Server allows parties sending microphone data to communicate | Abhishek Bhardwaj | April 4th, 2014 **(not started)** |
| Server Queries Database | Server queries database for song information and playlists | Shane Spoor | April 4th, 2014 **(not started)** |
| Final Testing | Rigorously test all features and document results | Shane Spoor, Ramzi Chennafi, Abhishek Bhardwaj | April 6th, 2014 **(in progress)** |
| Project Deadline | Project is fully tested and documented, all features are finalised | Shane Spoor, Ramzi Chennafi, Abhishek Bhardwaj | April 7th, 2014 **(in progress)** |