VNACT

Virtual Navigation And Communication Tool

Senior Design Team 14-22

The team...

Nicholas Schulze

Derek Petersen

Christopher Tedford

Charles Patterson

Client: VRAC at Iowa State. Funded by the United States Army.

Advisor: Dr. Stephen Gilbert

Project Plan Outline

1. Problem Statement
2. Project Concept
3. Requirements
4. Risks and Mitigations
5. Milestones

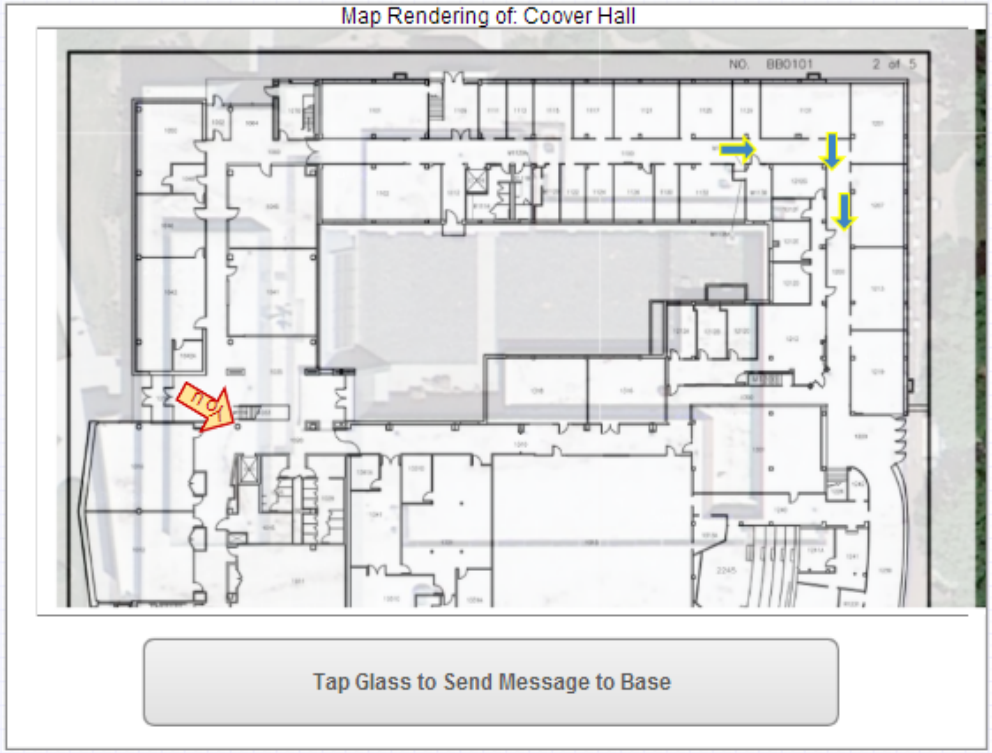
Problem Statement

Aid in the navigation and communication of Army soldiers conducting training exercises using the latest available technology.

Our goal is to develop a Google Glass application that provides an Army combatant with:

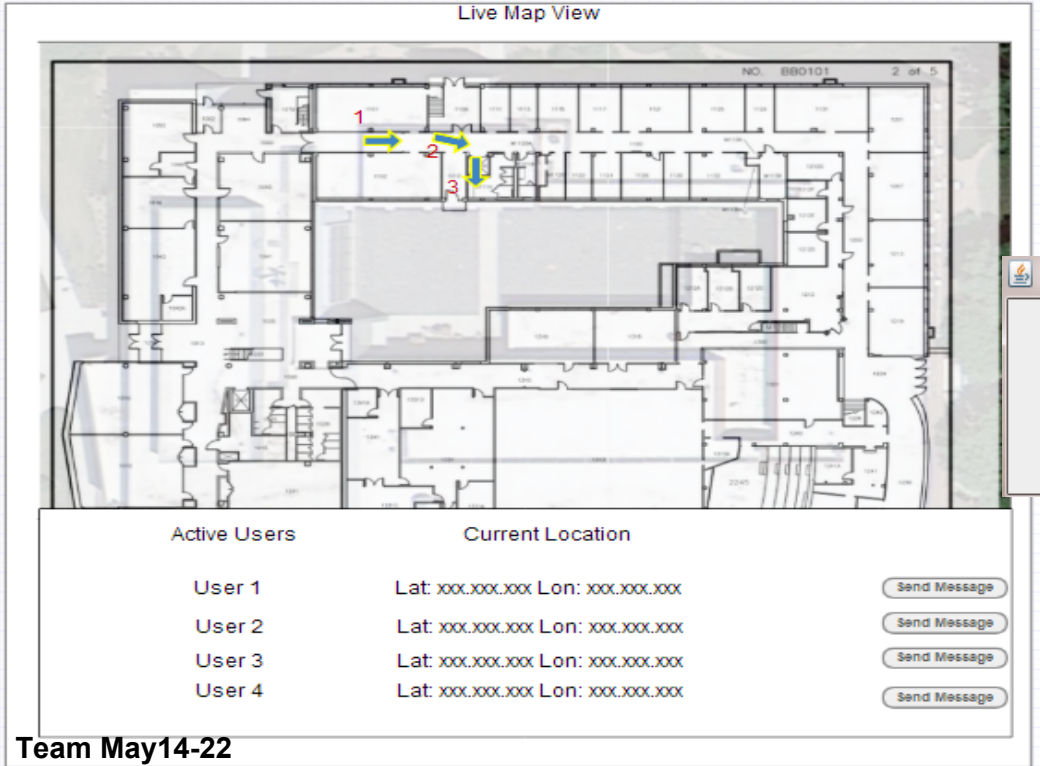
- **Dynamic real-time indoor/outdoor navigation**
- **Two-way audio communication with base location (server)**
- **Live video feed from the combatant to the base location (server)**

Project Concept: Glass Application (Client)

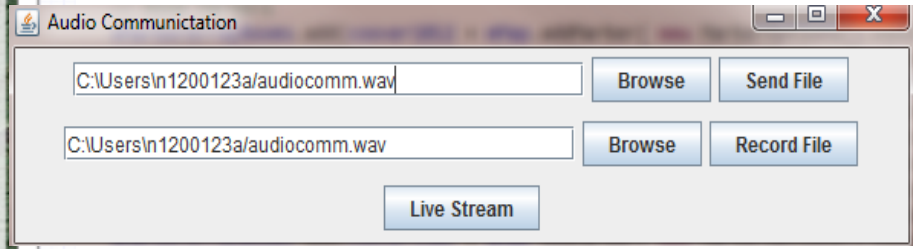


Current Android Prototype

Project Concept: Server Application (Base)

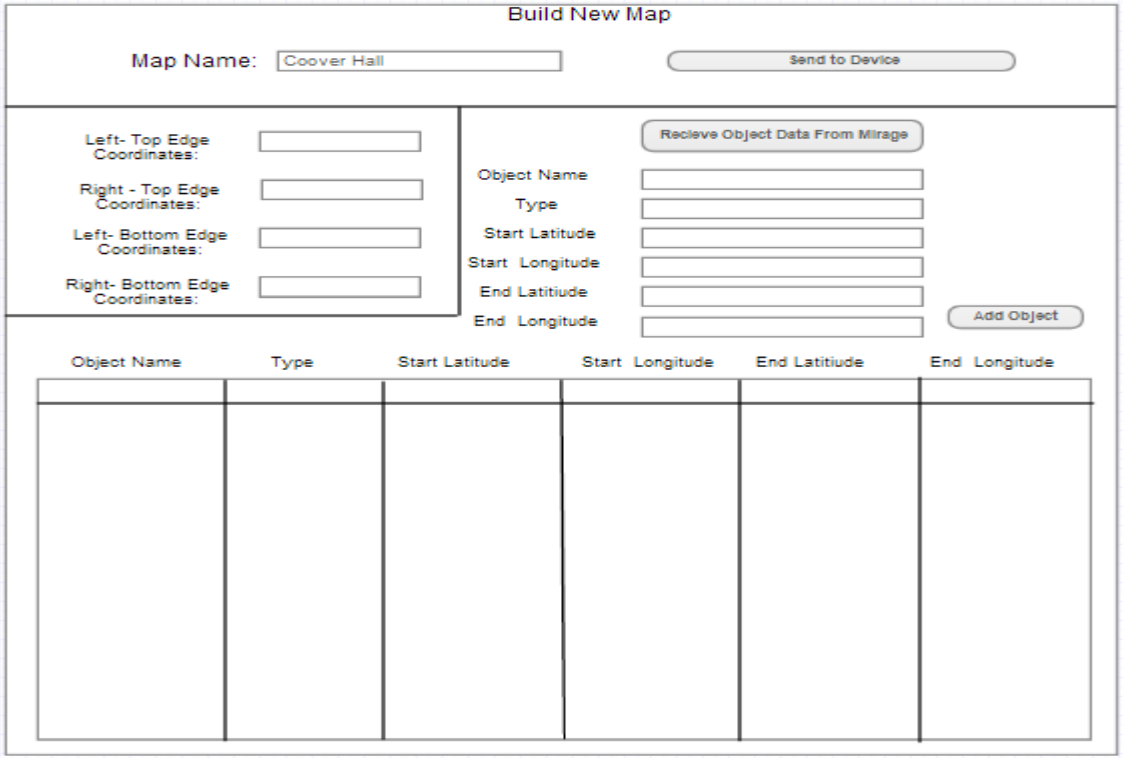


Server side application that acts as the base for the Army combatants.



Current Java Prototype

Project Concept: Server Application (Base)



Options for adding a new map to the Google Glass application:

-Recieve data from an external source and build map from location data (MIRAGE location data)

-Take image of a drawing/ picture of a map and load it as an overlay

-Add map from a file

-Hard code object locations from server

Major Requirements

Functional	Non-functional
● Display user's current location on top-down map along with all other users registered with the system within 500ms of user's actual location. ● Facilitate 2-way concurrent audio communication between the server and all users registered with the system. ● Stream video from all users registered with the system to the server where it will be displayed.	● The location, audio, and video data should be sent, received, and handled all within 500ms. ● The tracked location is accurate up to 5 feet. ● The location display should be completely independent of the location data source.

Technical Constraints

Battery Life: Google claims a day of use, we have found otherwise.

Mirror API: Have been using the standard Android API.

Glass API: Recently released, will build the final product using this in the Spring.

Indoor GPS Functionality: Currently an issue, we will use the Mirage tracking system.

Glass Software Limitations: Currently Google Play Services (Google Map API) is not working on the Google Glass system. Google states they are in the progress of getting Play Services to work.

Risks and Mitigations

Risk	Probability	Criticality	Risk Factor	Mitigation Strategy
Google Glass cannot meet our goals for the project	.25	70	.25 * 70 = 17.5	Research hardware and APIs.
The Google Glass is damaged, broken, or stolen and can no longer be used in development.	.25	100	.25 * 100 = 25	Set up a system to ensure nothing happens to the Glass and attempt to acquire another.
The development team is incapable of developing software to meet the requirement	.05	70	.05 * 70 = 3.5	Allow developers time to become comfortable with the relevant software and hardware.

Risks and Mitigations

Risk	Probability	Criticality	Risk Factor	Mitigation Strategy
We can't effectively track the user indoors, using GPS.	.25	70	.25 * 70 = 17.5	Leverage the tracking system in the Mirage to get the user's location.
The Google Glass API is changed, affecting the functionality of the app.	.10	60	.10 * 60 = 6	Root the Glass or find a work around that allows us the functionality we want.
The development team can't complete the project on time because they are too busy.	.05	70	.05 * 70 = 3.5	Ensure that each developer sees this project as a priority.

Milestones

Fall 2013

1. Finalize Project Requirements
2. Create Plan and Schedule
3. Create Design
4. Prototype Audio Communication and Map Display Functions

Milestones

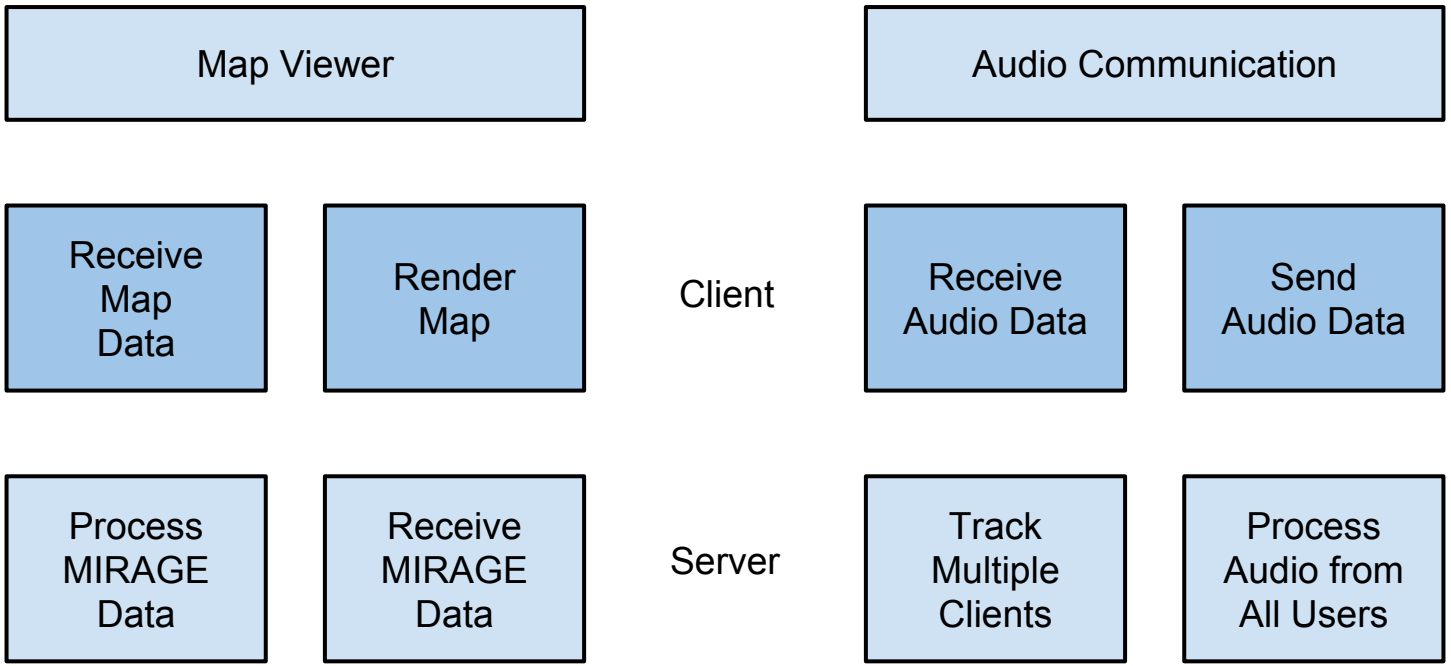
Spring 2014

1. Write and test server-side implementation for a single registered client.
   a. Complete map data handler for MIRAGE
   b. Complete audio stream handlers
   c. Complete video stream handler
   d. Complete integration testing
2. Write and test client-side implementation
   a. Complete map display for MIRAGE
   b. Complete 2-way audio stream
   c. Complete video stream
   d. Complete integration testing
3. Scale server up to handle 8 registered clients.
4. Complete implementation of Google FloorPlan map data handler

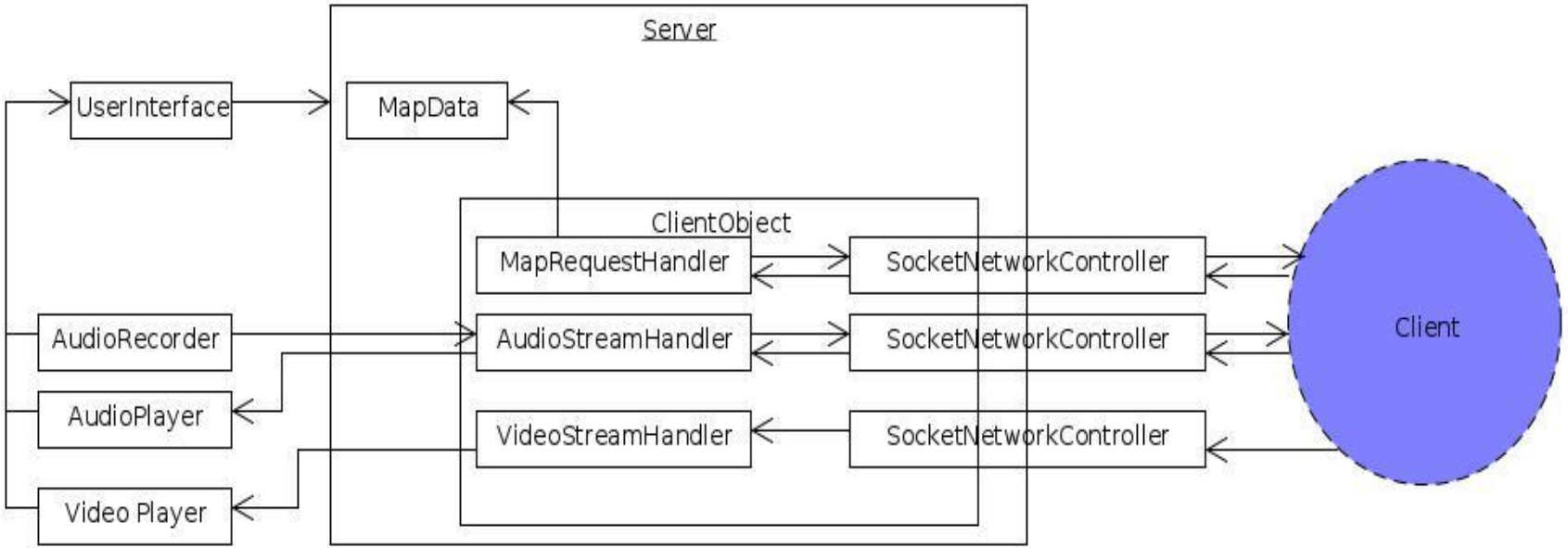
Project Design Outline

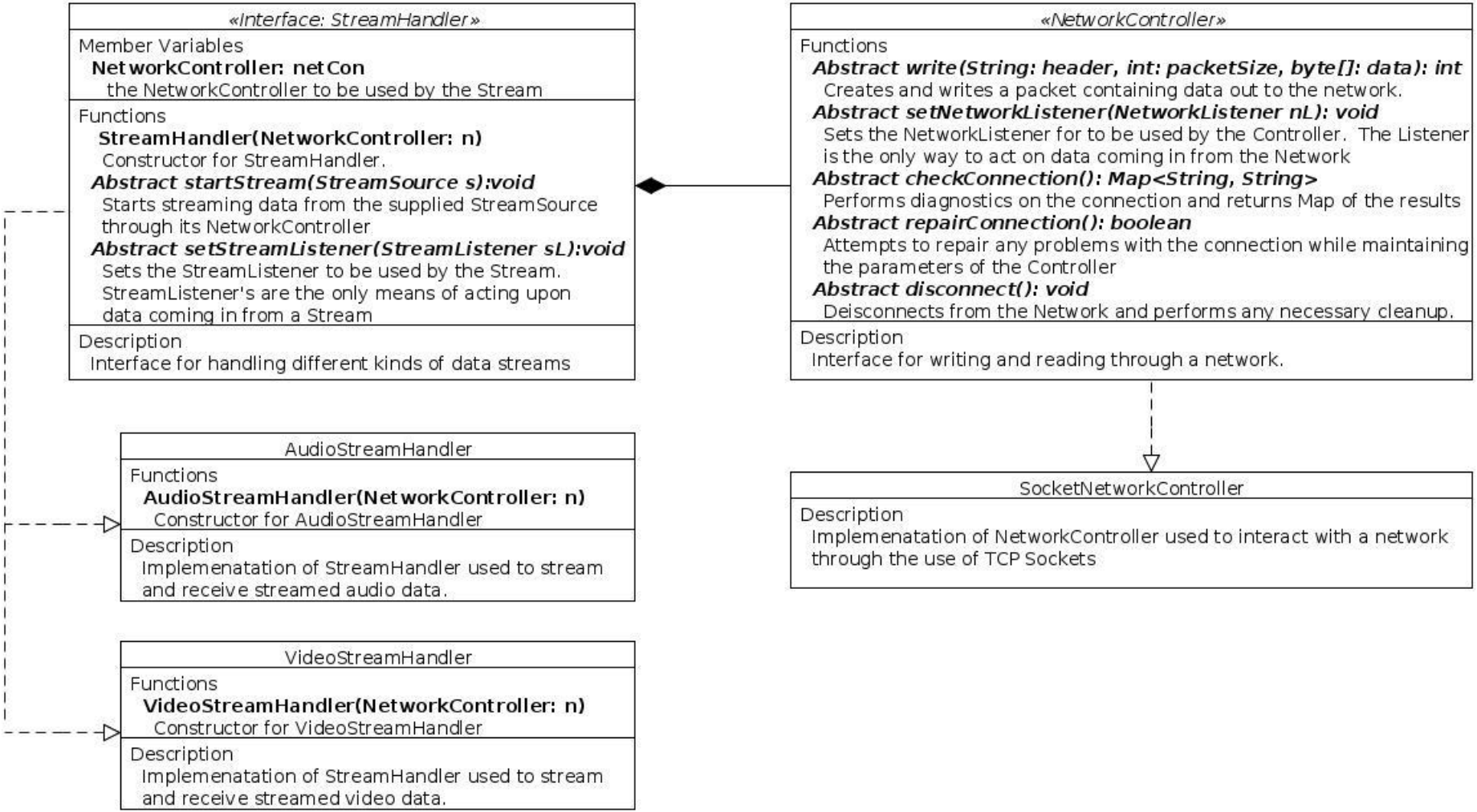
1. Functional Decomposition
2. Detailed Design
3. Test Plan
4. Prototype Demo

Functional Decomposition



Server Interaction Diagram




«Interface: StreamHandler»
Member Variables
NetworkController: netCon
the NetworkController to be used by the Stream
Functions
StreamHandler(NetworkController: n)
Constructor for StreamHandler.
Abstract startStream(StreamSource s):void
Starts streaming data from the supplied StreamSource through its NetworkController
Abstract setStreamListener(StreamListener sL):void
Sets the StreamListener to be used by the Stream. StreamListener's are the only means of acting upon data coming in from a Stream
Description
Interface for handling different kinds of data streams
«NetworkController»
Functions
Abstract write(String: header, int: packetSize, byte[]: data): int
Creates and writes a packet containing data out to the network.
Abstract setNetworkListener(NetworkListener nL): void
Sets the NetworkListener for to be used by the Controller. The Listener is the only way to act on data coming in from the Network
Abstract checkConnection(): Map<String, String>
Performs diagnostics on the connection and returns Map of the results
Abstract repairConnection(): boolean
Attempts to repair any problems with the connection while maintaining the parameters of the Controller
Abstract disconnect(): void
Deisconnects from the Network and performs any necessary cleanup.
Description
Interface for writing and reading through a network.
AudioStreamHandler
Functions
AudioStreamHandler(NetworkController: n)
Constructor for AudioStreamHandler
Description
Implemenatation of StreamHandler used to stream and receive streamed audio data.
SocketNetworkController
Description
Implemenatation of NetworkController used to interact with a network through the use of TCP Sockets
VideoStreamHandler
Functions
VideoStreamHandler(NetworkController: n)
Constructor for VideoStreamHandler
Description
Implemenatation of StreamHandler used to stream and receive streamed video data.

Test Plan

Based on our agile development workflow.

We will make incremental changes to the project over time, adding features and fixing bugs.

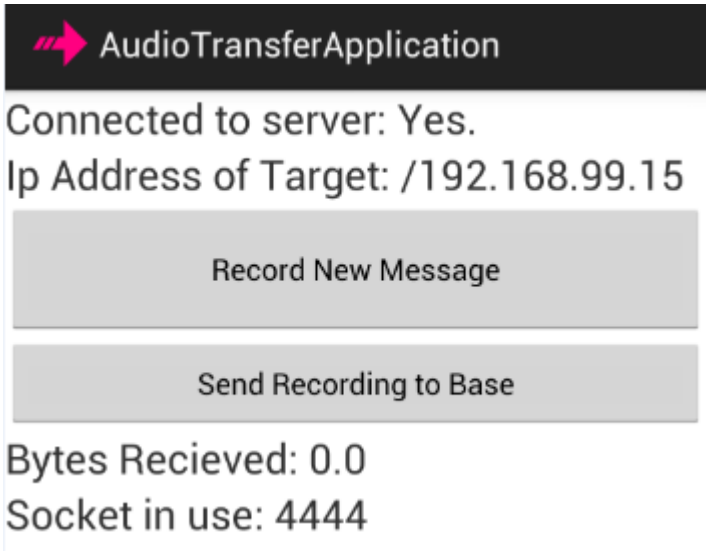
For each new module our agile process is as follows:

1. Design Component
2. Test Component
   2.1 Usability Testing: All functional and nonfunctional requirements of module as well as measures testing.
   2.2 Software Mock Tests: Trial implementation of module in main application with blackbox and whitebox testing.
3. Repeat steps 1 and 2 until satisfactory
4. Implement Component

Example: Adding audio transfer over TCP from client to server

First Prototype Demo

Demo of Audio Application on Google Glass.



Timeline for Spring 2014

Task	Start Date	End Date
Develop Network Sections of Server and Client sides	January 13th	February 3rd
Develop Map, Audio, Video Sections of Server and Client sides	February 3rd	March 14th
Start Integrating Server and Client sides	March 14th	March 28th
Start scaling system up to handle 8 simultaneous users	March 28th	April 11th
Start Final Testing/Integration	April 11th	May 2nd
Prepare for Final Review/Demo	May 2nd	May 9th