TalkSee

Design Document

Team 6

Mark Higa
Chris Bauschka
Ashley Braun
Aaron Petry

Table of Contents

Purpose

TalkSee is a Windows Phone application that utilizes the Microsoft Research Hawaii SDK and a wireless internet connection. The application is designed to aid people who have hearing conditions or may simply be in an environment where it is difficult to hear the phone speaker. As users communicate through a voice or video call over an internet connection, the app will provide speech-to-text captions for spoken words, providing a visual means and supplement for communication. Users will also be able to manually type and send text communications during the call.

Summary of Functional Requirements

User Experience

- Clients can connect to the server to see who is available to call.
- Clients can change their username and server connection port.
- In a call, clients can see video from the other client.
- In a call, clients can hear voice from the other client.
- In a call, clients can see the transcribed text of the voice.
- In a call, clients can send chat messages to each other.

Directory Server Responsibilities

- The server will run indefinitely.
- The server will keep a list of connected clients.
- The server will keep a list of available clients.
- The server will keep a list of busy clients.
- The server will push out the lists to all connected clients whenever there is a change in status.
- The server will remove clients that timeout or disconnect.

Speech-to-Text Cloud Responsibilities

- The STT server will run indefinitely.
- The STT server will accept voice packets.
- The STT server will translate the voice packets into text.
- The STT server will return the translated text to the client.

General Priorities

The design decisions detailed in this document are based on the following priorities, listed in decreasing level of importance.

Usability

The primary focus of our application is to ensure that it is intuitive as possible for users will all levels of expertise to be able to easily navigate and use our product. It should be easy for new users to quickly pick up how to use our application. The user interface should be simplistic and intuitive, free of unnecessary clutter. The speech-to-text feature should add to a conversation, not detract from it.

Reliability

It is key that both our application and our server be reliable. The directory server should have minimal downtime and should be resilient to bugs and errors. The speech-to-text server, controlled by Microsoft, should always be able to provide its services to our application. The application itself should also reflect these qualities, such as not crashing and handling effectively.

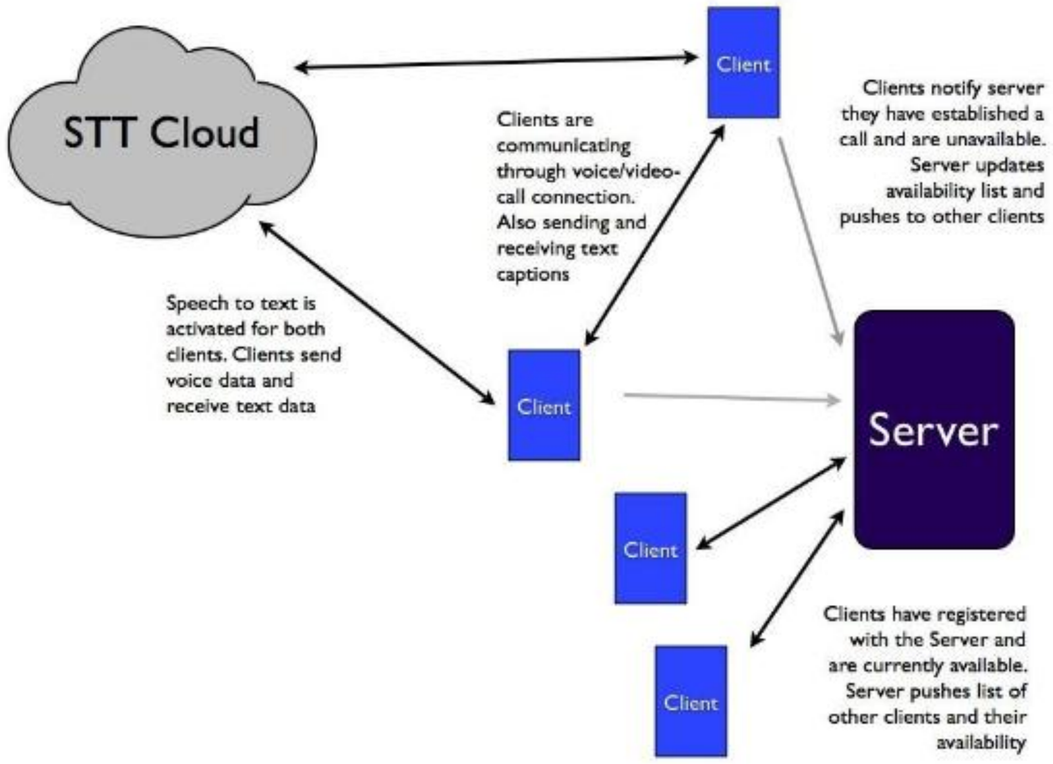
Efficiency

In accordance with the nature of mobile applications, our application should be designed to minimize battery drain, optimize processing time, reduce memory allocation, and reduce network utilization.

Performance

The both the directory and STT server should be able to scale up to support multiple users and calls at the same time.

Outline

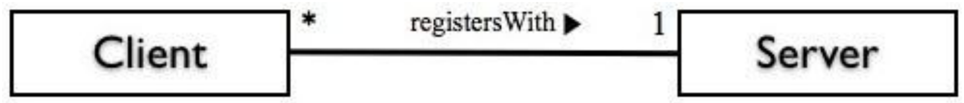

STT Cloud
Client
Client
Client
Client
Server
Clients are communicating through voice/video-call connection. Also sending and receiving text captions
Clients notify server they have established a call and are unavailable. Server updates availability list and pushes to other clients
Speech to text is activated for both clients. Clients send voice data and receive text data
Clients have registered with the Server and are currently available. Server pushes list of other clients and their availability

The TalkSee application will utilize a client/server architecture. The server will communicate with the client application on Windows Phone devices through an internet connection. The client will also communicate with Microsoft's speech-to-text (STT) Cloud service when text captions are activated.

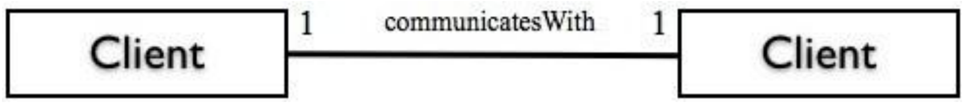
Each client will establish a connection with the server and send information about itself which the server will store. Each client will keep the server updated with its availability and connection status which the server will store and share with all other connected clients.

An available client can initiate a call to another available client and begin voice/video-call communication. If speech-to-text captions are activated, the currently speaking client will continually record and send voice clips to the STT Cloud. After translating the speech to text, the STT Cloud will send the text back to the speaking client, which will immediately send it to the currently listening client. Both clients will store the sent and received text as a transcript while they are connected.

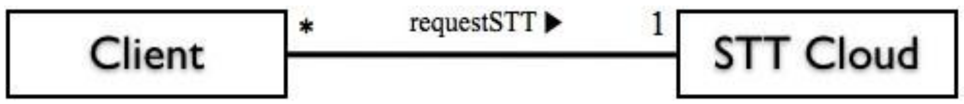
UML Diagrams



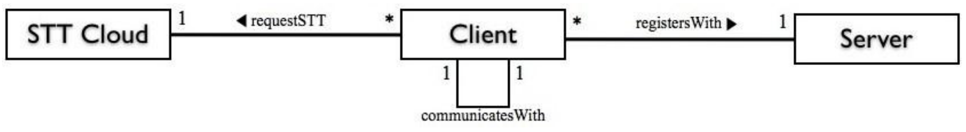
Multiple clients can register with Server.



Each client can only connect and communicate with one other client.



Multiple clients may request speech-to-text translations from STT Cloud



Multiple clients can register with Server. Each client can only connect and communicate with one other client. Multiple clients may request speech-to-text translations from STT Cloud.

Major Design Issues

Platform

Options:

1. Windows Phone
2. Android
3. IOS
4. other

Decision: Windows Phone

We decided to go with the Windows phone because it offered to us the ability to use the Microsoft Research Hawaii SDK, which has the speech-to-text translation feature that is key to our application. It also offers the possibility of easily translating the application into a program that runs on the Windows 8 OS.

Windows Phone Version

Options

1. 7.5
2. 8.0
3. other

Decision: 7.5

While the 8.0 version of the Windows Phone OS has many exciting features, we are limited by the availability of testing equipment. Therefore, we chose to use the 7.5 version as a balance between features and availability.

Server Implementation

Options

1. Python
2. C/C++
3. Java
4. C#
5. other

Decision: Python

We decided to use Python to implement our server because of the reduced development time and rapid prototyping. Additionally, we are using a relatively simplistic communication protocol, so it is not necessary to use a different language with a more rich protocol interaction library.

Python Version

Options

1. 2.6
2. 2.7
3. 3.2

Decision: 2.7

While Python 3 has a rich set of features and is recommended for future development, it is not as widely integrated as earlier versions are. For that reason, we have chosen Python 2.7 as the most optimal for our project because of its widespread integration and future compatibility.

IP Protocol

Options

1. TCP for all communications
2. UDP for all communications
3. Selectively use both

Decision: Selectively use both

We decided to use a combination of TCP and UDP protocols for our project. The voice and video communications will use UDP to reduce lag time and closer synchronization. The communications with the directory server and the text transmission will use TCP to guarantee reliable and error free messages.

UI Design

Options

1. Design a unique user interface with our own unique stylistic guide
2. Follow Window's Phone's styling and that of similar applications

Decision: Follow Window's Phone's styling and that of similar applications

We feel that providing a familiar interface will improve user experience, streamline the flow of the application, and maintain the unique Window's Phone experience that drew users to this particular mobile operating system.

Interclass and Class/Server Communication

Options

1. Class abstraction
2. Single class

Decision: Class abstraction

We decided to abstract the dialer functionality out to another class to streamline the pieces of communication, and to simplify the implementation of the project. The user class also helps aid object creation, so that we can treat users as objects. The TalkSee class also aids in handling the GUI and the directory server two-way communication.

User Settings Persistence

Options

1. Store the settings of a unique client on the server and send them to the client when it registers
2. Store the settings locally on the phone

Decision: Store the settings locally on the phone

We decided to store the setting on the phone because that would reduce stress on the directory server and simplify the directory server implementation. Additionally, this would be more in line with the functionality of most other apps.

Client-Server Communication

Options

1. HTTP
2. other

Decision: other

We will implement our own simplistic communication protocol using TCP to facilitate the exchange of the directory listing between the directory server and each client. This will also be used by the client to notify the server of the status of a client.

User Identification

Options

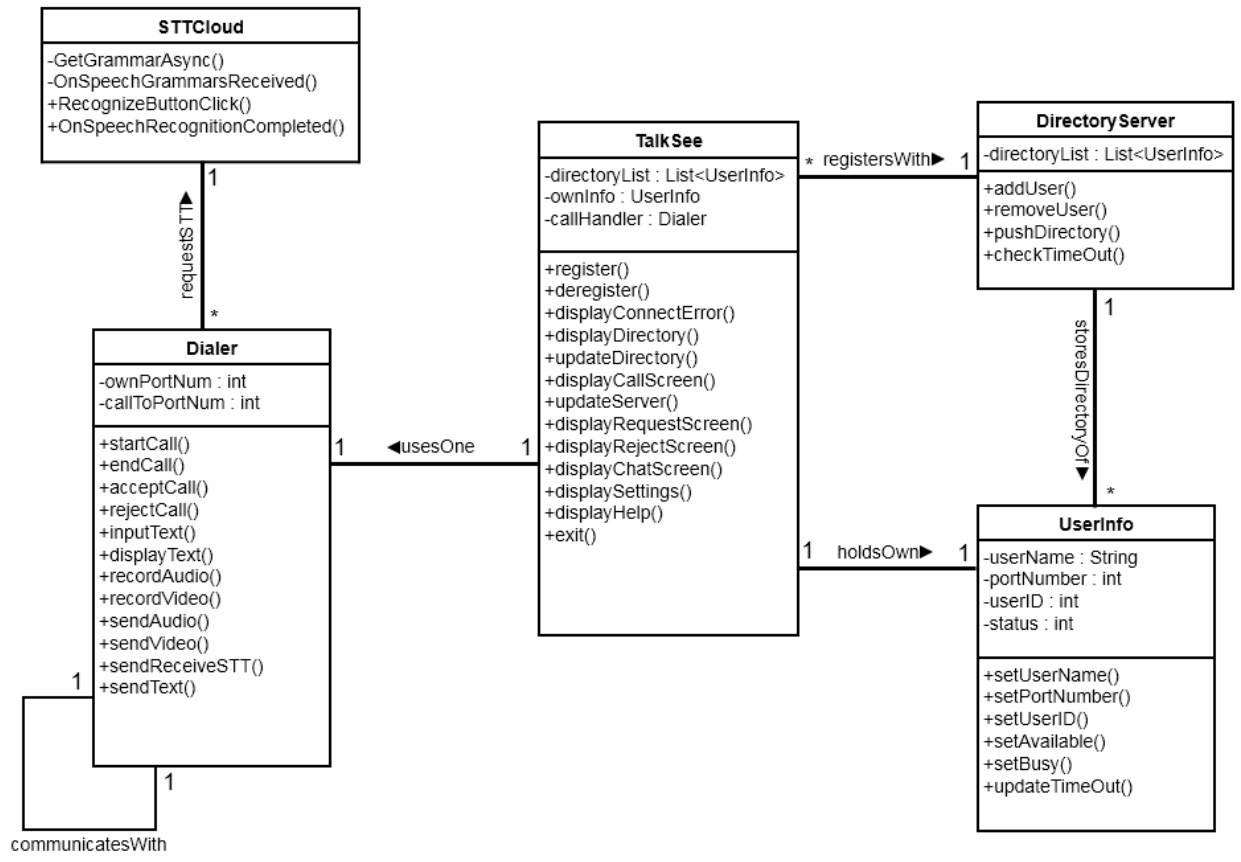
1. Login
2. No Login

Decision: No Login

We decided to not require users to log into the system because of time constraints. This can be implemented in the future, and we will design the underlying architecture to support future development in this area.

Details of the Design

Class Diagram



Description of classes and models

TalkSee

The main phone application driver that handles registering and updating the DirectoryServer, the phone's graphical user interface, and its Dialer. It also holds a UserInfo object with its own information and a DirectoryList received from the DirectoryServer.

DirectoryServer

Entity that holds and manages a List of UserInfo objects of the current registered phones.

UserInfo

Entity that encapsulates all user information and provides methods for managing it.

Dialer

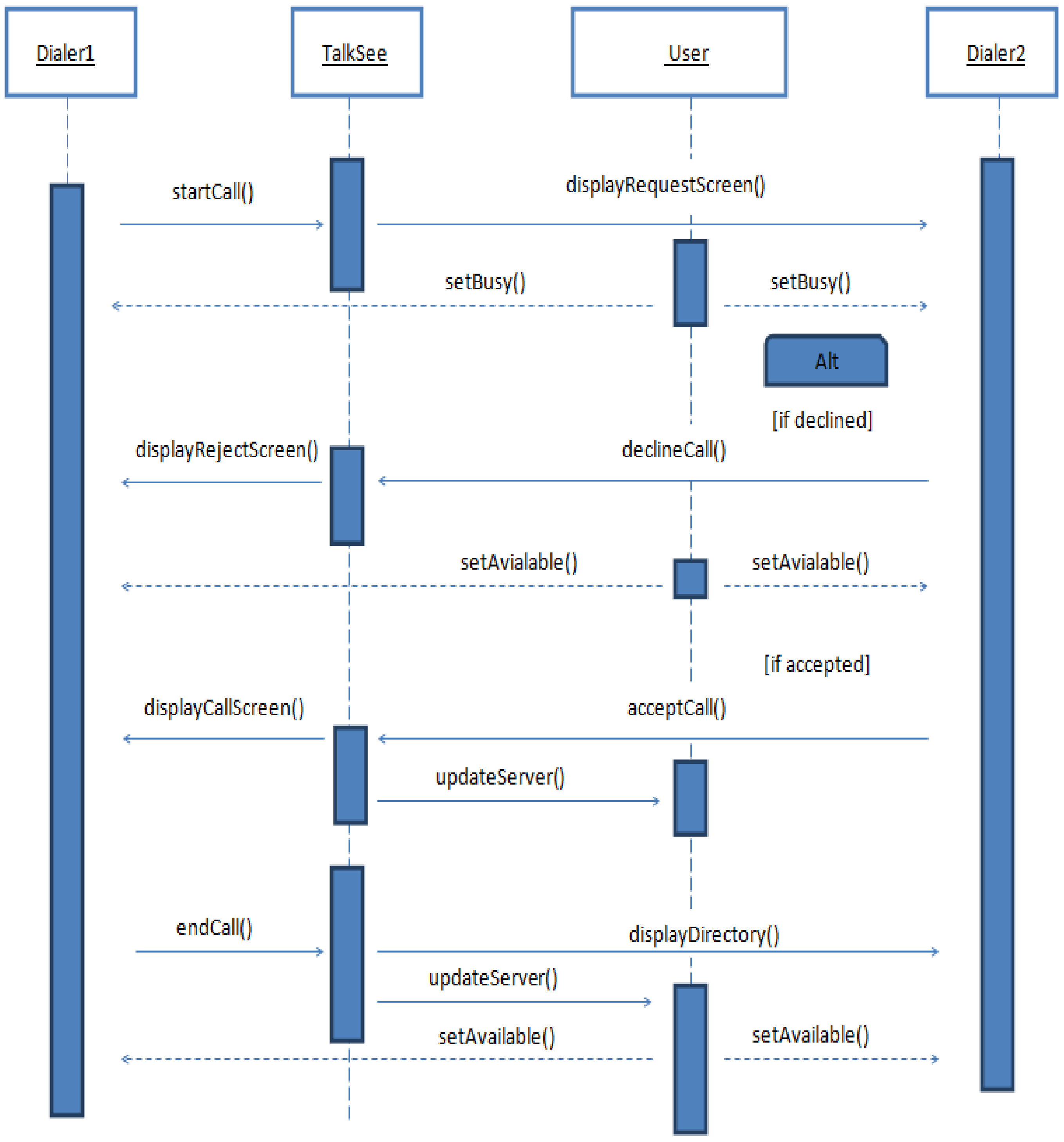
Entity that handles all communication functions. It can communicate with one other Dialer at a time. It also handles communication with the STTCloud to send audio clips and receive text translations.

STTCloud

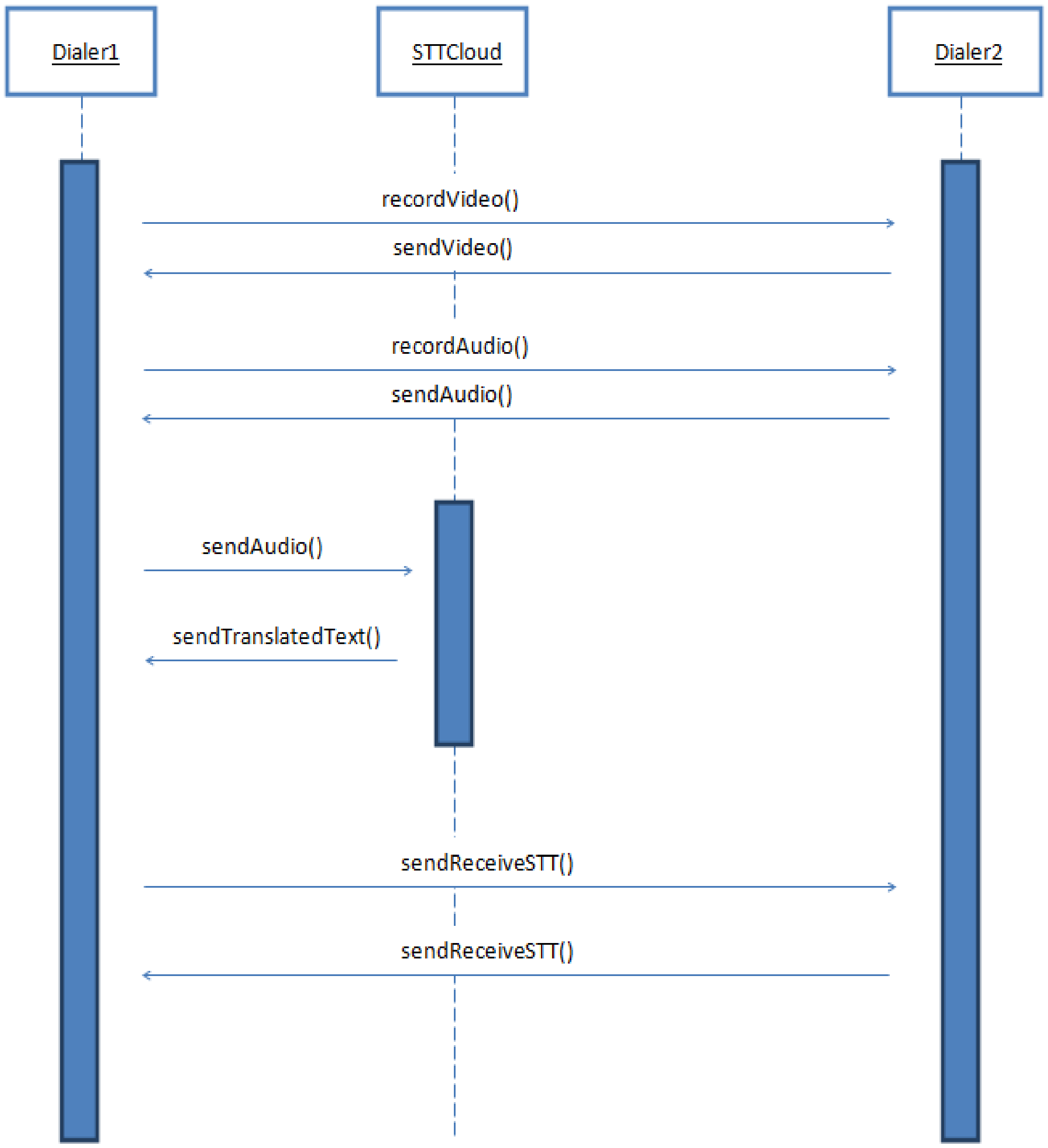
Entity provided by the Microsoft Research Hawaii SDK that handles audio to text translations. Additional details that do not pertain to directly to our system are left out for simplicity.

Sequence Diagrams

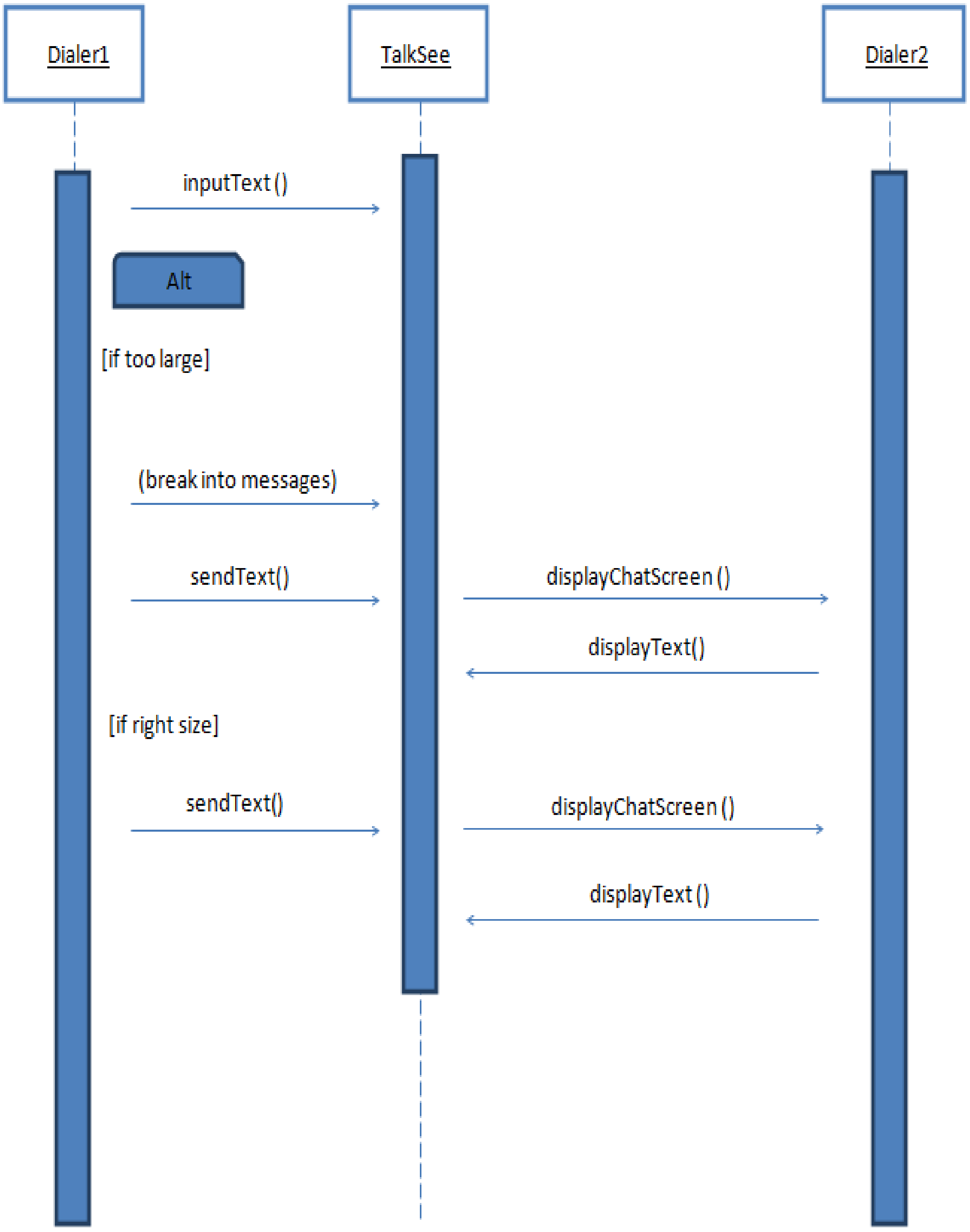
Starting and Ending Calls



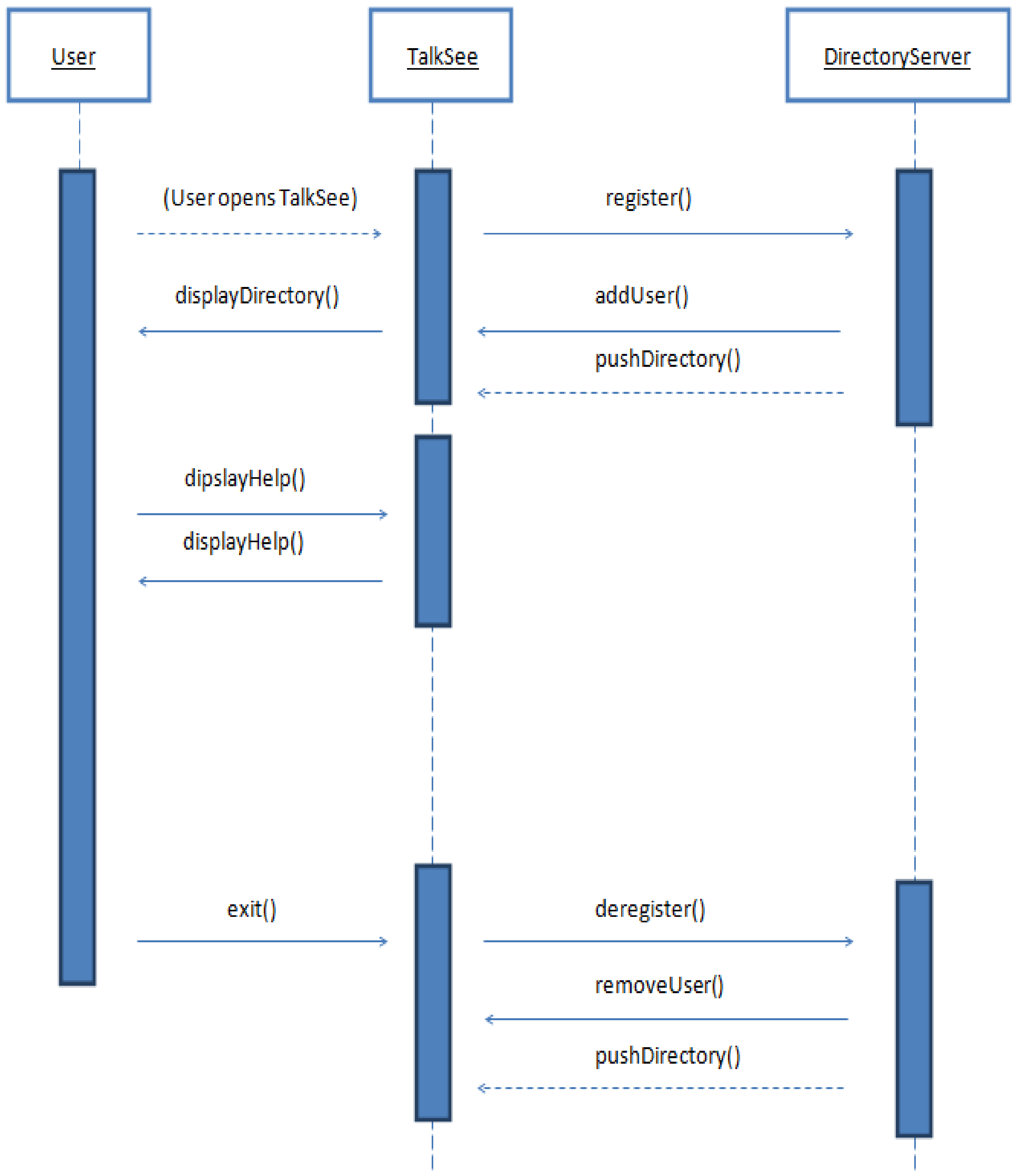
Sending and Recording Audio and Video, Sending Speech-To-Text



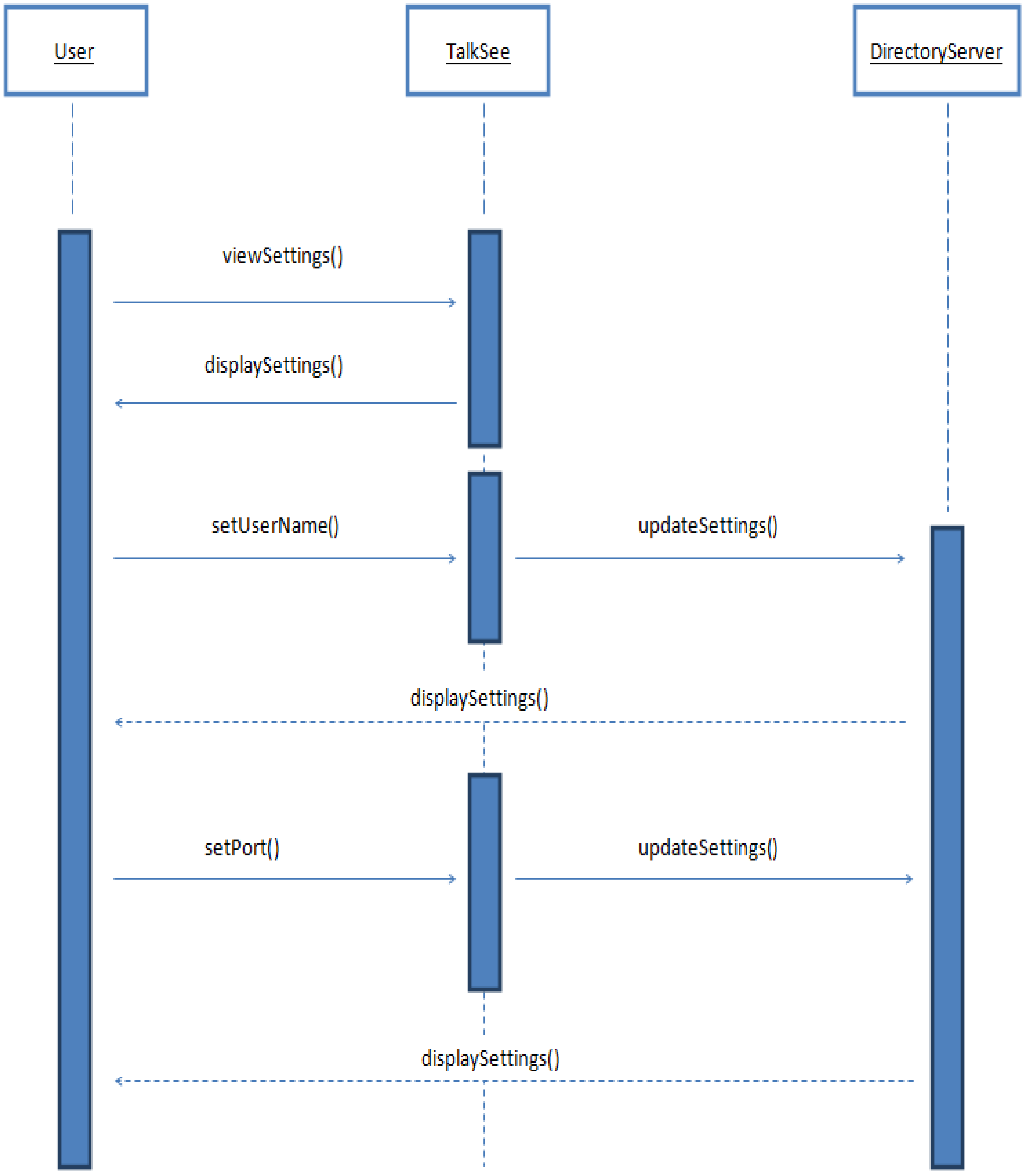
Sending Chat Messages



Opening TalkSee, Displaying Help, Closing TalkSee

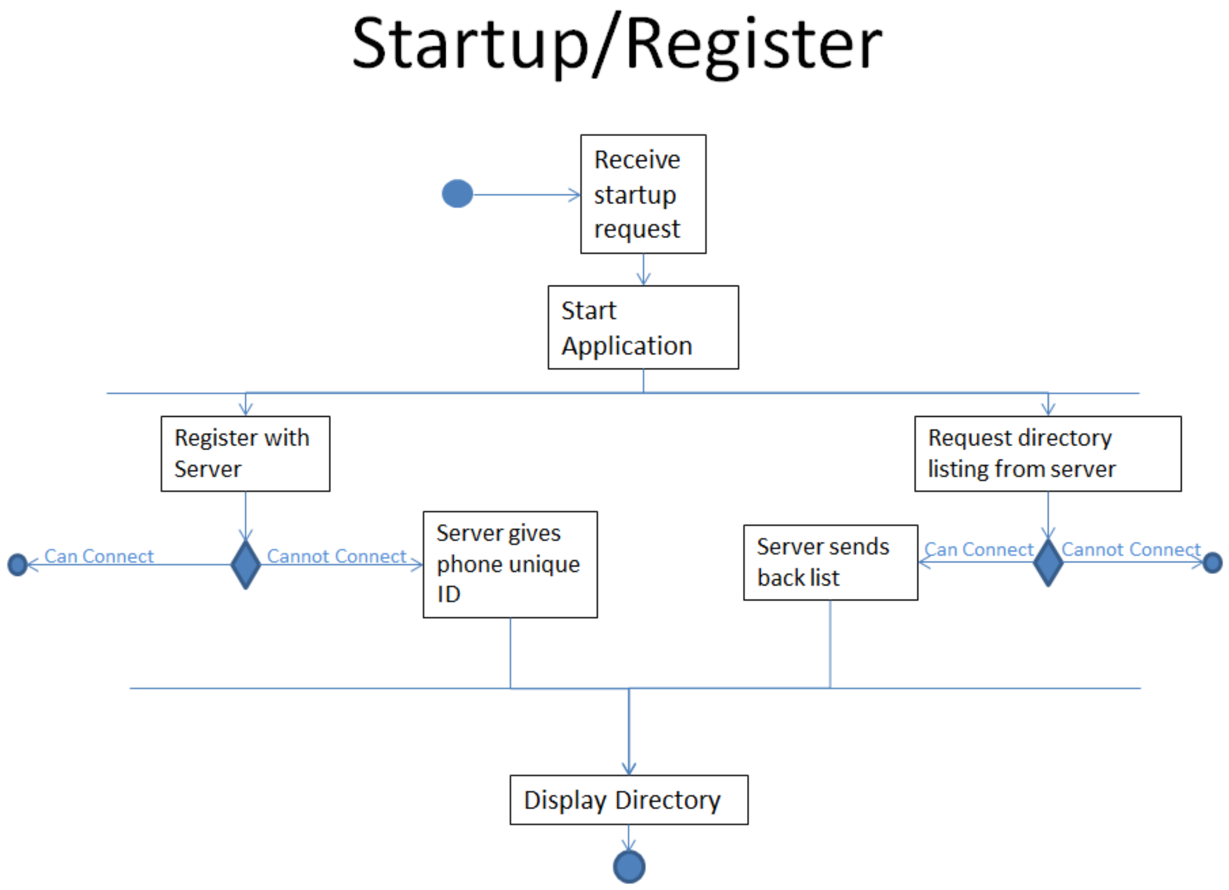


Changing the Client Settings (Port and Username)



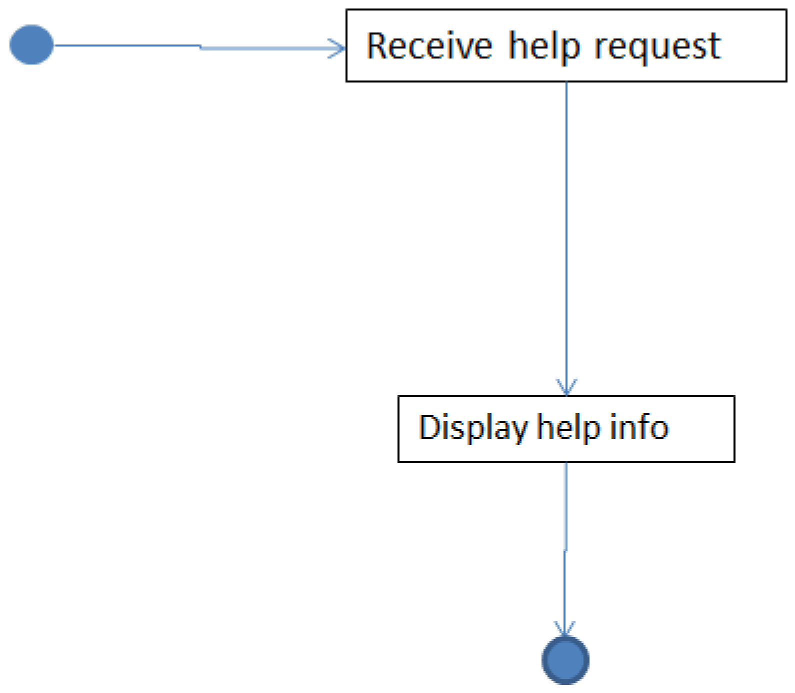
Activity Diagrams

Startup / Register



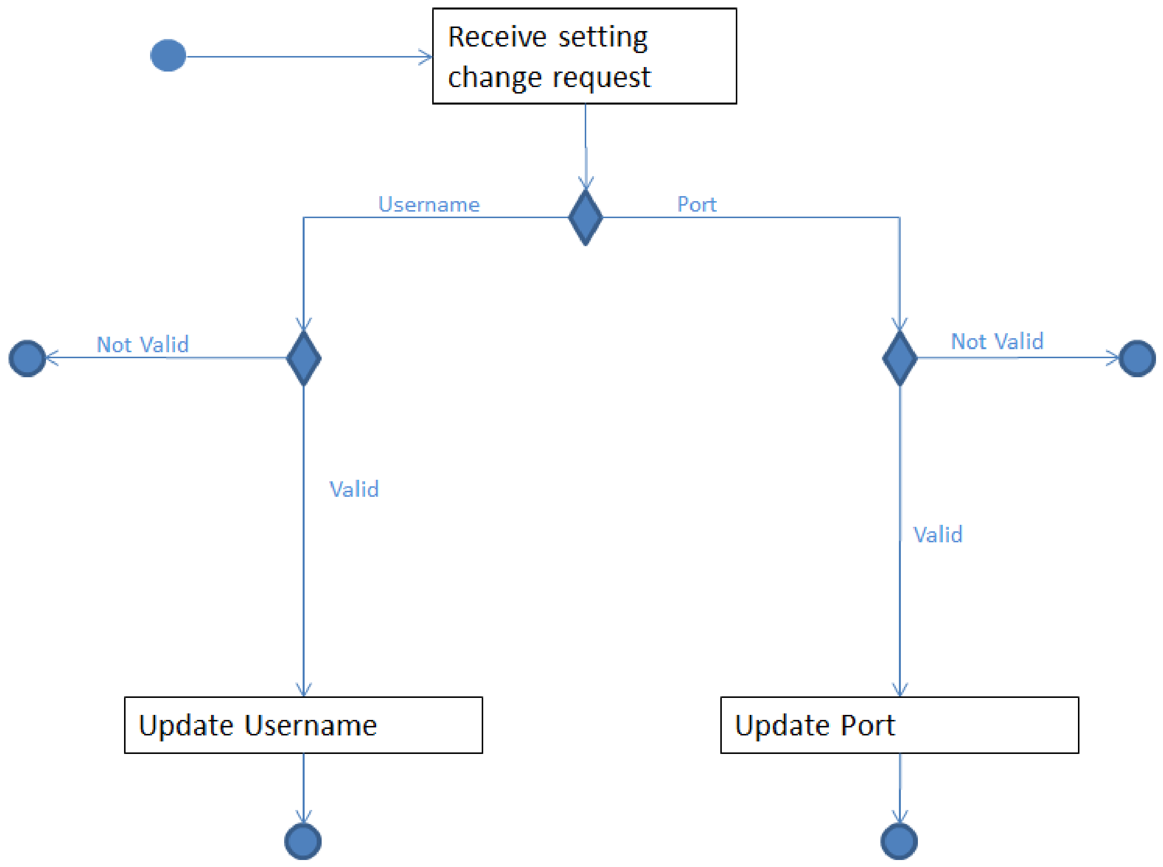
When the user requests to start the application by selecting it from the phone's list of apps, the application will start up, then will both register with a server to get a unique id, and request the directory listing from the server. If both of these complete successfully, the directory will display.

Help



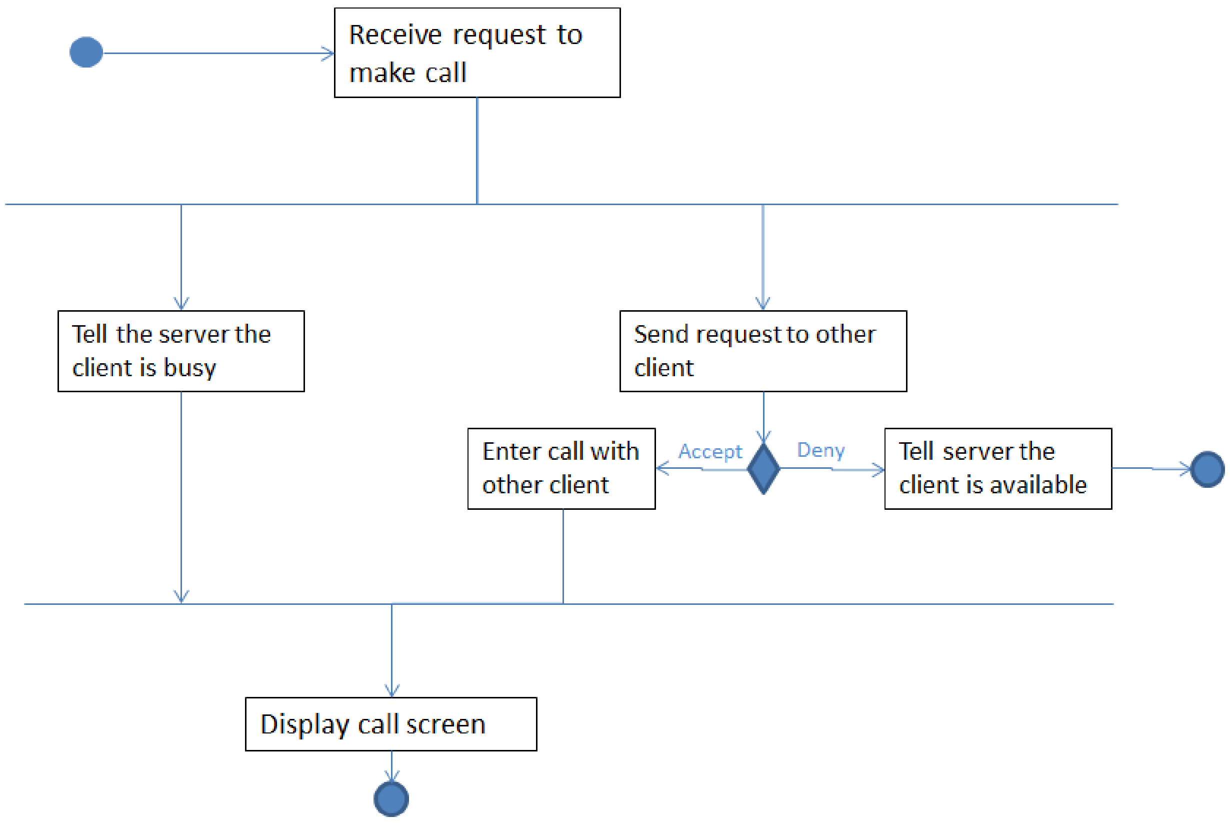
When the user selects help from the main screen, the help screen will display.

Settings



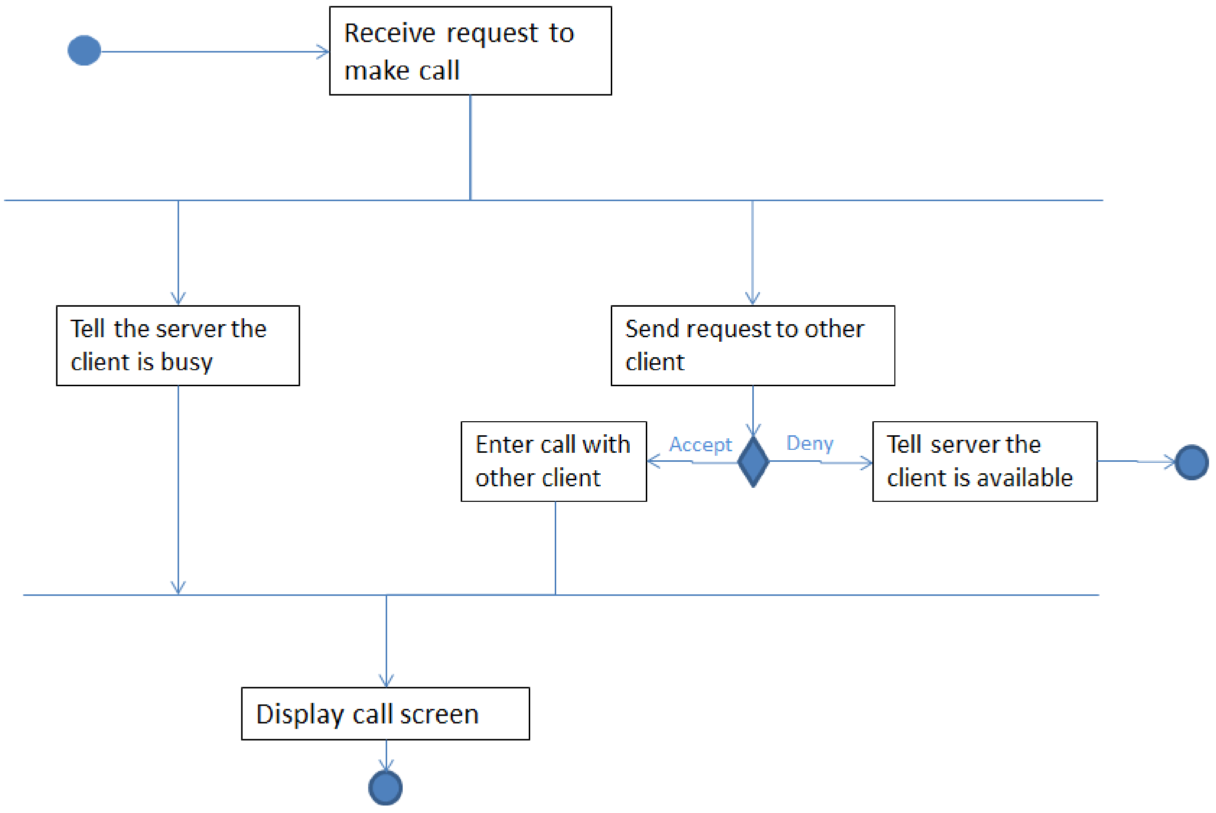
When the user requests to change the setting, it will first check to see whether the username or the port is being changed. Then, it will check for the validity of the requested update. If it is valid, the new value will be stored on the server.

Make Call


Receive request to make call
Tell the server the client is busy
Send request to other client
Enter call with other client
Accept
Deny
Tell server the client is available
Display call screen

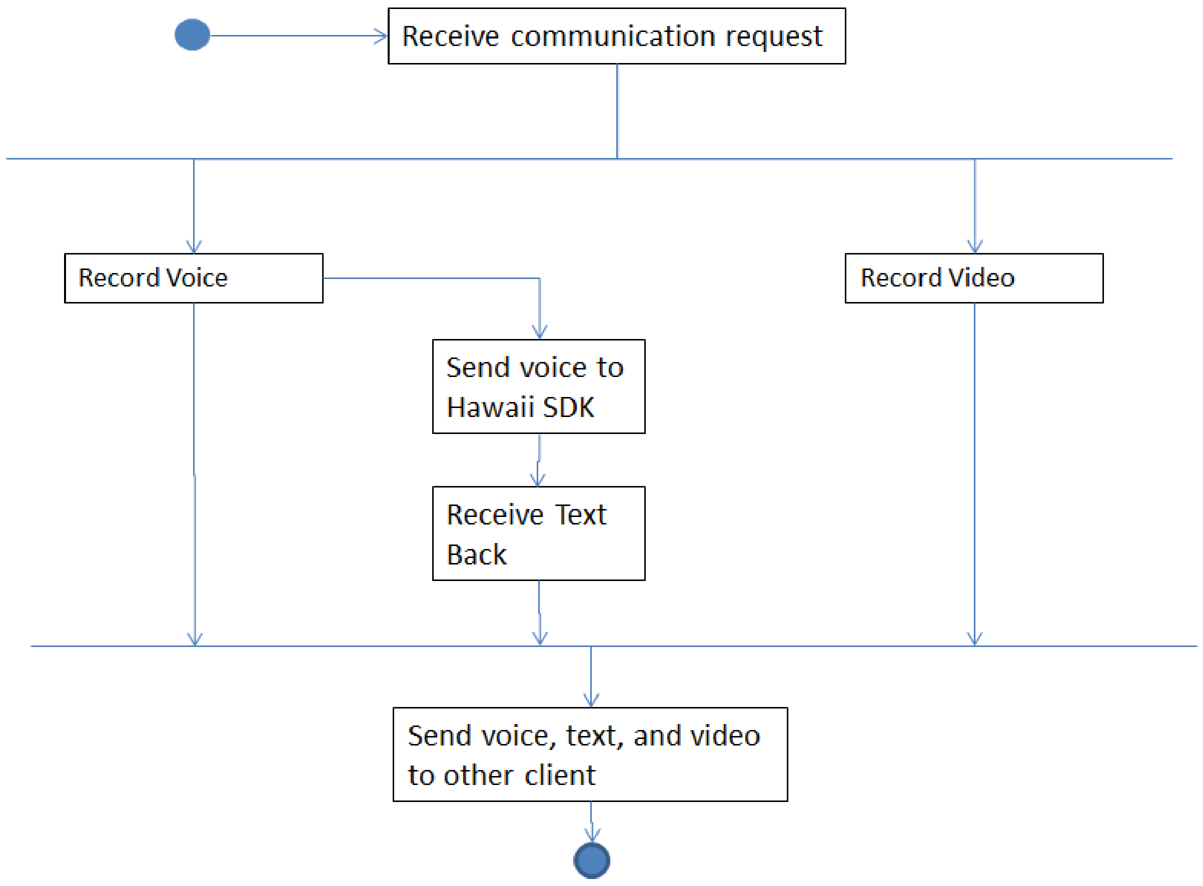
When the user asks the application to make a call, the client will tell the server that it is busy, and it will send a request to the other client, who will either accept or deny. If they accept and the server knows they are busy, the call screen will display. If the other client denies, then the client will become available again to the server.

Receive Call


Receive request to make call
Tell the server the client is busy
Send request to other client
Enter call with other client
Accept
Deny
Tell server the client is available
Display call screen

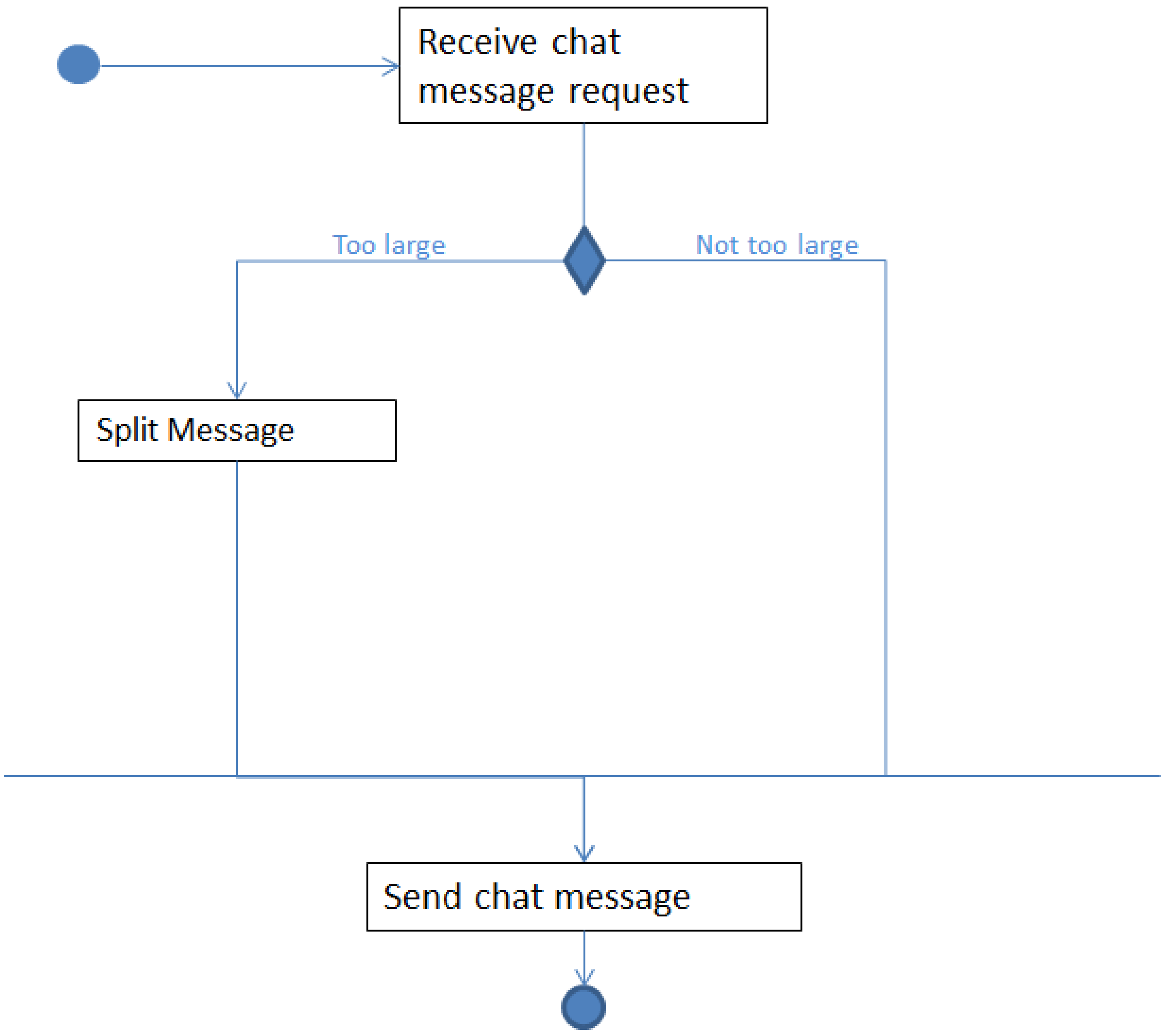
When the application receives a request to enter a call from another client, the application will tell the server that it is busy, and it will display a decision box to the user, who will either accept or deny. If they accept and the server knows they are busy, the call screen will display. If they deny, then the client will become available again to the server.

Communication


Receive communication request
Record Voice
Record Video
Send voice to Hawaii SDK
Receive Text Back
Send voice, text, and video to other client

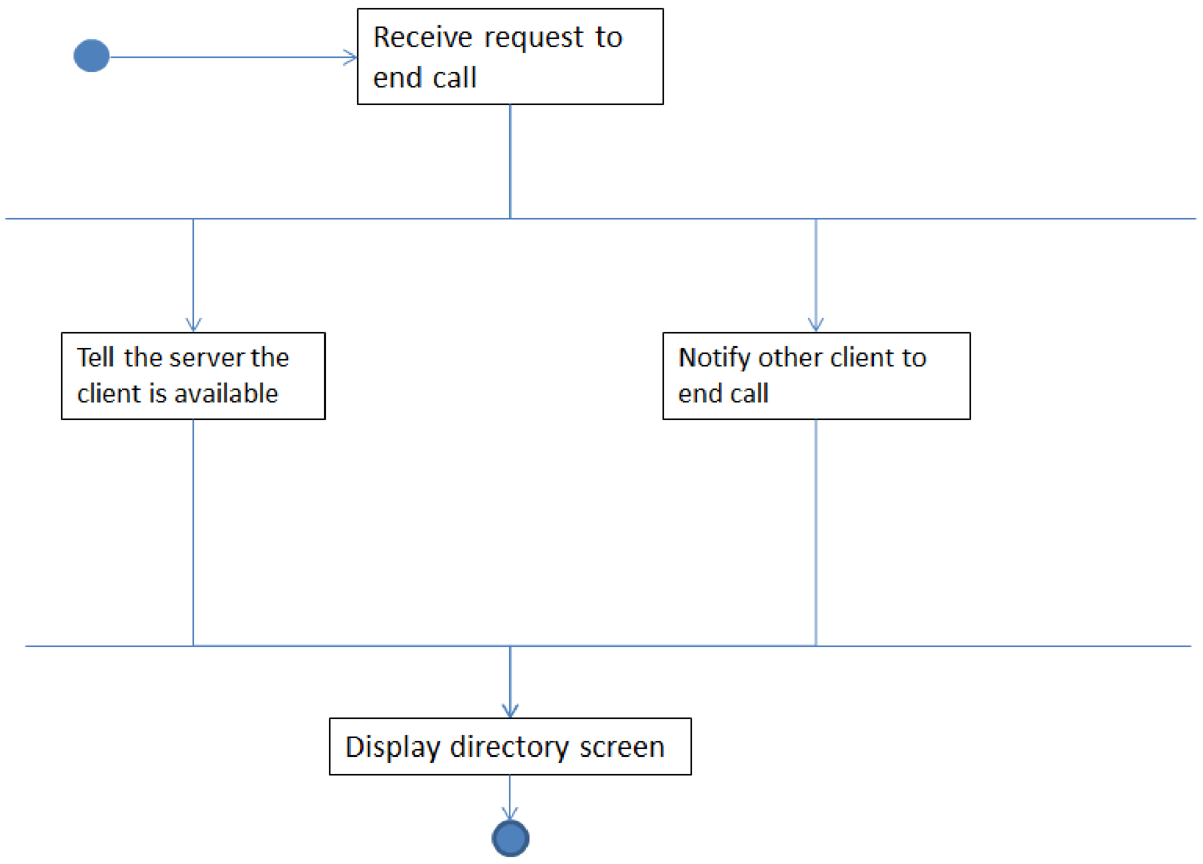
Continuously throughout a call, the application will be asked to record voice and video. It will take the voice and send it to the Hawaii SDK, which will send back text. Once the application has voice, text, and video, it will send all of them over to the other client.

Chat Message



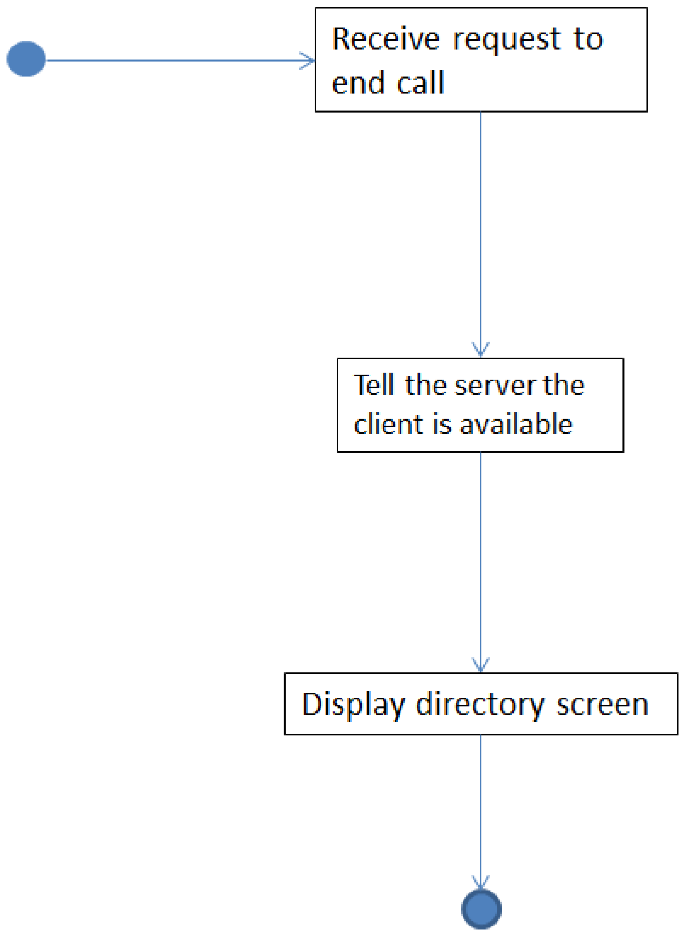
When the application receives a request from the user to send a manually typed chat message, it will first check to make sure that the message does not exceed the buffer boundaries. If it does, then it will split the message into manageable pieces. It will then send the chat message to the other client.

User Ends Call


Receive request to end call
Tell the server the client is available
Notify other client to end call
Display directory screen

When the application receives a request from the user to end the call, it will notify the server that it is now available, and it will notify the other client that the call is being terminated. Afterwards, the directory screen will display again.

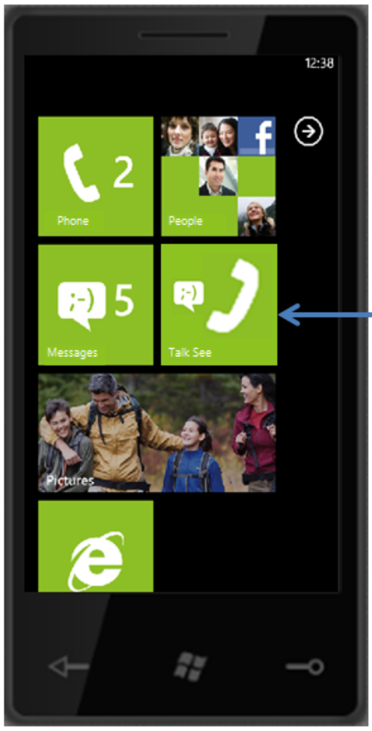
Other User Ends Call


Receive request to end call
Tell the server the client is available
Display directory screen

When the application receives the message that the call was terminated by another client, it will tell the server that it is available, and then it will display the directory screen.

UI Design Mockups

Application List

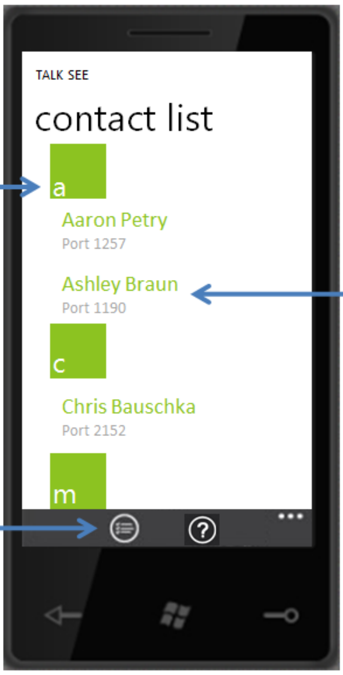


TalkSee will display in the list of applications, and can also be pinned to the start screen of the phone as seen here

Home Page

Clicking on a letter will allow the user to navigate quickly to any other letter

Help and settings buttons are located at the bottom. Clicking the three dots to the right will expand the icons to show labels


TALK SEE

contact list

a

Aaron Petry
Port 1257

Ashley Braun
Port 1190

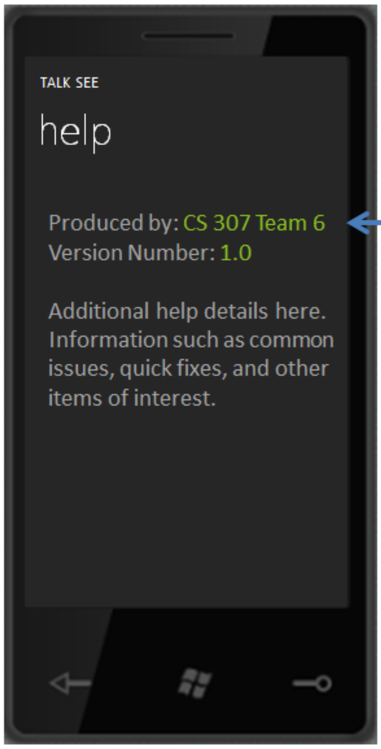
c

Chris Bauschka
Port 2152

m

The user will be taken directly to a contact list, with the names of everyone on the server. Clicking on a name will call that person.

Help



TALK SEE

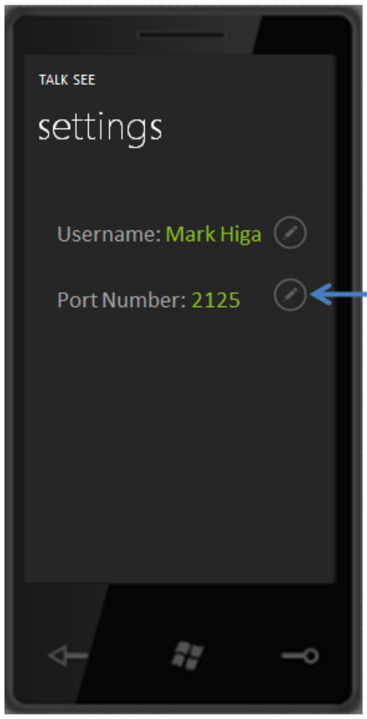
help

Produced by: CS 307 Team 6
Version Number: 1.0

Additional help details here. Information such as common issues, quick fixes, and other items of interest.

Help will be an informational page displaying the producing organization, the version number, and any other details that the user would be interested in.

Settings



TALK SEE

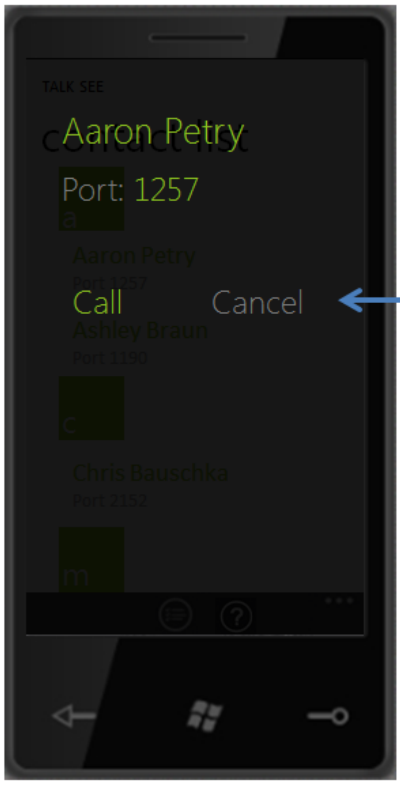
settings

Username: Mark Higa

Port Number: 2125

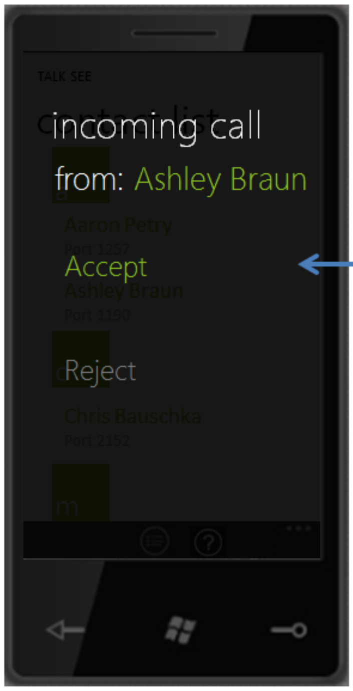
The user will be able to change their username and port number here by clicking on the symbol next to the field

Call Client



This screen will display after a username is clicked. They will be able to call that person, or cancel the call

Incoming Call

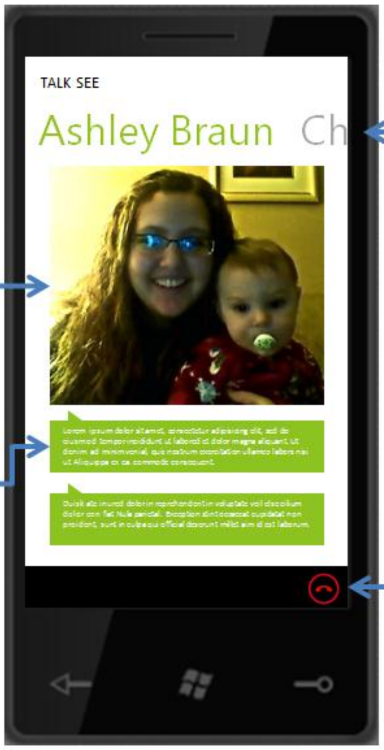


This screen will display when an incoming call is registered. The user will be able to accept or reject the call

In Call

This screen will display when a user is in a call. They will be able to see video of the other user

Text of the other user's speech will be displayed here, along with any messages they typed in themselves

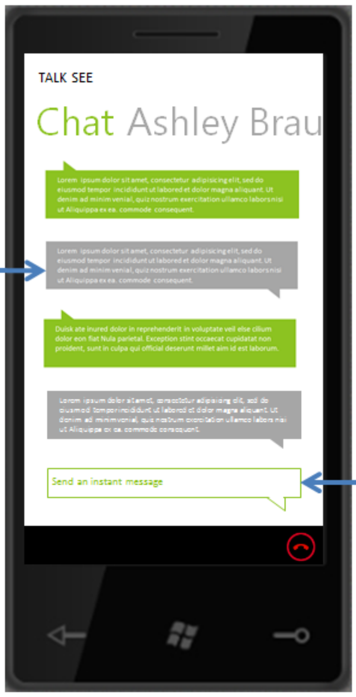


Menu headings at the top will display the other users name, as well as options to swipe to the chat screen

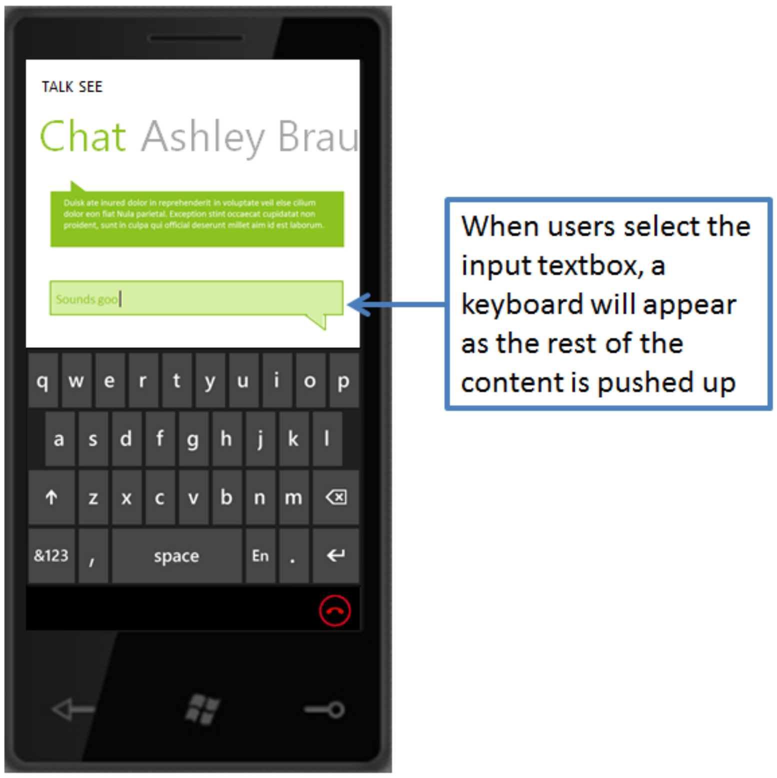
Users will be able to end the call using this button

Chat

Swiping to the right from the video screen will allow the user to arrive at this screen, where they can view past messages, and type messages in manually to send to the user. The call will still remain active.



Users can enter custom text here



Duisk ate inured dolor in reprehenderit in voluptate veil else cilium dolor eon fiat Nula parietal. Exception stint occaecat cupidatat non proident, sunt in culpa qui official deserunt millet aim id est laborum.

Sounds goo

When users select the input textbox, a keyboard will appear as the rest of the content is pushed up