M2M WISE VIDEO STREAMING APPLICATION REQUIERMENTS DOCUMENT

COMPANY Deutsche Telekom

TECHNICAL ADVISOR MR. DUDU MIMRAN

ACADEMIC ADVISOR DR. BRACHA SHAPIRA

TEAM MEMBERS

ASSAF SHEMESH

HILA VISAN

ILYA GIDALEVICH

IOSEF FELBERBAUM

TABLE OF CONTENT

1. INTRODUCTION

1.1 - VISION

The main goal of our project is to create a platform that enables video communication between two standard mobile smart-phones. We intend to enable live video streaming between the two phones without having to pass through a "middle man" server that deals with the session initiation, the video transfer or the massive computations.

We are about to deal with communication problems, overcome obstacles like various network conditions (i.e. congestion), the quality of the arriving stream.

We will also be able to conform to the existing network infrastructures: WIFI, 3G.

Our platform will enable one-way video communication for usages like: virtual studying classes, supervision of kindergartens, nurses, nannies etc.

After the problems mentioned above are handled, our project may easily be expanded into many other usages including two way video communication, text and voice integration etc.

1.2 – THE PROBLEM DOMAIN

Today, video streaming is everywhere – video calls between cell phones, live streaming of movies on the internet and more. This technology is needed in various aspects of life, from security needs to entertainment. Its necessity is extremely important in our everyday life, and is the main subject constructing our project's problem domain.

As the main subject of our problem domain, video calls between two smart phones define two sub-problems: **Video calls on Android operating system**, and **Video calls based on P2P model**. This hierarchy describes the problem domain, which derives concrete problems that our project implementation should answer.

Implementing an application over the problem domain described above, faces some complex problems that need to take in account while implementing - **technical problems**. While constructing the project, the implementation should supply appropriate solutions to these problems.

We will focus on these three following technical problems:

- **Transferring big amount of information in real-time through a cellular network**

  Streaming of video and sound between cellular smart-phones requires transferring large amount of data between the phones. This data is transferred through the network, but has to be transferred fast enough, to keep the sense of real-time communication between the parties. Transferring the big amount of data through the network takes time, especially in cellular networks like 3G which is slower than WIFI network.

  Also, transferring video and sound through the network requires encoding and decoding the information. This requires time, and can also slow the process.

  Altogether, transferring large amount of data through the cellular network and dealing with encoding and decoding processes affect the communication between the parties, make the communication slower – and damage the real-time sense required from the conversation.

- **Maintaining good quality video and sound streaming in real-time between cellular platforms**

  In developing an application for video streaming, our goal is to supply high-quality communication between the parties. However, there is a tradeoff between video and voice quality and speed of transfer the data from one party to another. The higher the quality of voice and video, the larger the size of data to transfer. Therefore, it is necessary to consider reducing video quality for improving data transfer speed. The problem is – how much do we wish to reduce in sound and video quality for better transfer time and real-time communication, but still provide reasonable video quality, which can be viewed and understood by the human eye.

- **Defining flexible protocols for communication between two cellular smart-phones, connected on different networks – 3G and WIFI**

  As described before, two important aspects of the program is the speed of transferring the data through the network and the quality of services provided from it – video and audio. While supplying high quality video and audio transmission through the network, the cell-phones connectivity to the network raises another problem – cellular smart-phones can be connected through different network infrastructures. In the same connection, one party can be connected through 3G connection, and another party can be connected through WIFI connection. It is known that 3G connection is slower than WIFI connection, and this can affect the quality of conversation between the parties.

  It is necessary to provide a good solution for this problem. A good one is to define flexible protocols that can handle transition from one network to another, without the overhead it requires. By defining these protocols, faster transitions between the networks is possible by their flexibility, and the quality of service provided by the application is not damaged.

As for the Major-Components Diagram of the larger system, this diagram is not relevant for this application we are about to develop. This application is designed as an Add-On for the Android OS, and not as a stand-alone application.

1.3 - STAKEHOLDERS

- Customers:

  The main customers of the platform we are trying to develop are telecommunications companies, especially *Deutsche Telekom*, which would like to integrate our proposed functionality into their cell-phones.

  Companies will be interested in the project in order to attract more customers interested in such services.

- Direct Users:

  The direct users are actually the customers of the telecommunications companies, which we develop the application for. The users that will be interested in the functionality we enable vary between private users with personal needs (i.e. fun and entertainment, concerned mothers) and different organizations for their business purposes (i.e. military). Personal needs may be of fun and entertainment, supervising untrusted nursing services, little children etc.

  Organizations may find it useful to use it as a tool for interactive video-conferencing of high quality.

- Experts:

  The experts that might be involved in the developing process are *Video Experts*, *Mobile-Software Experts*, *Mobile-Hardware Experts*, *compression experts* and consults from networking fields of cell-phone industry.

1.4 – SOFTWARE CONTEXT

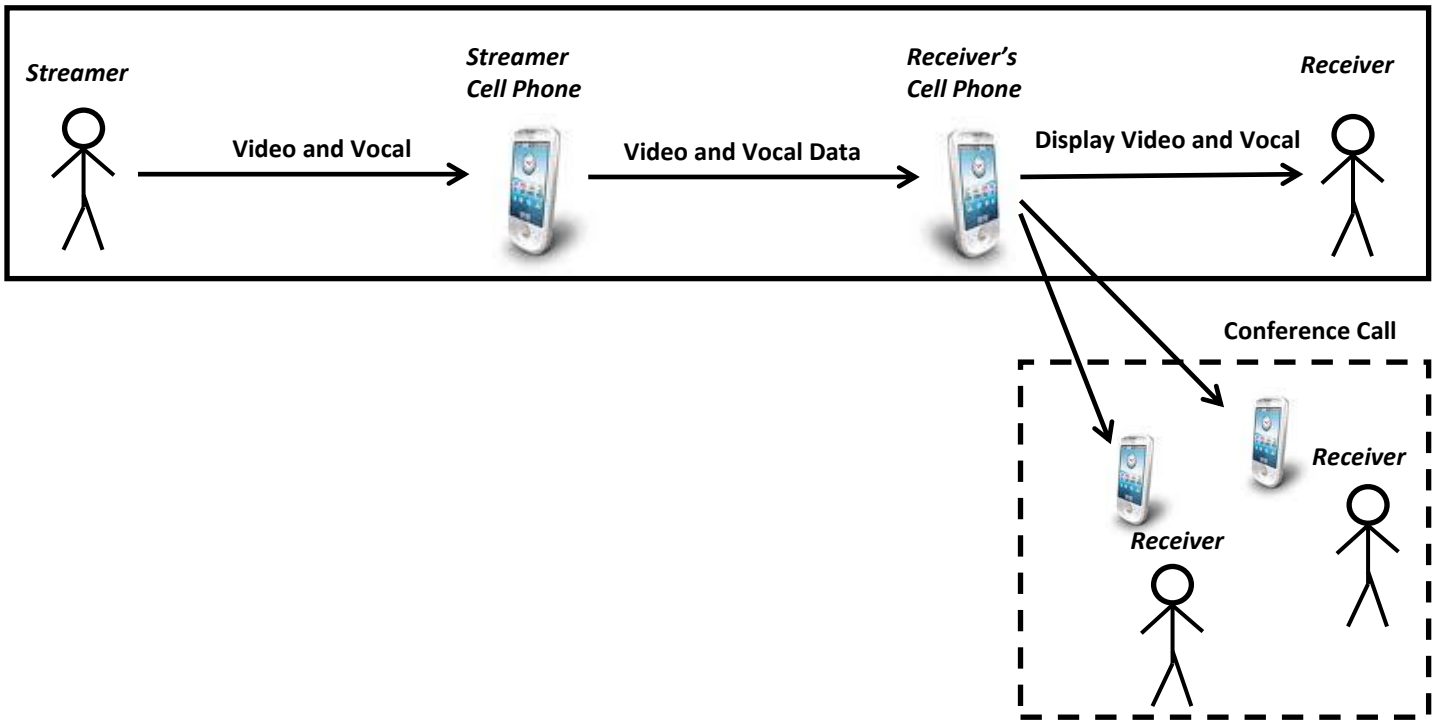


Figure 1 *– High Level Design Of The System.*

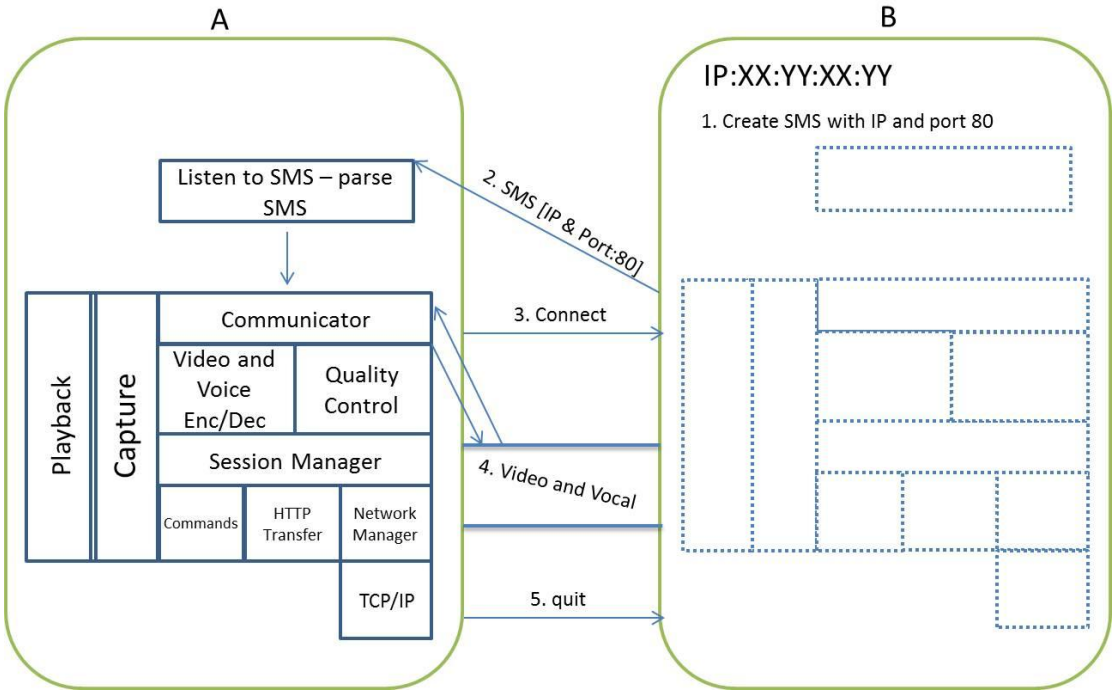


Figure 2 *– More Detailed Description of the System.*

1.4.1 Input

- **Streamer's Inputs:**
  - **Video and Vocal –** The Video and Vocal captured by the Streamer's cell phone.
- **Receiver's/Conference-Receiver's Inputs:**

  None.
- **Additional Inputs:**
  - **Phone Number –** The number of the desired contact to call to. Both sides can initiate a call using this input.
  - **Personal Settings –** The personal adjustments of the user for using the application.

1.4.2 Functionality

Streamer's Functionalities:

- **Start a call –** This functionality includes the session initiation: the phone's IP of the caller party will be sent by the application, using SMS message, to the callee party (selected by its phone-number). On the callee side, a service will sample the SMS inbox (***Listen to SMS – parse SMS*** in *figure 2*), recognize the special SMS message with the IP and port numbers, and notifies the callee on the incoming call.
- **Start capture video and vocal-** This functionality describes the situation in which a party becomes the streamer and starts streaming video and audio from his phone. Any participant of the call can become the streamer of the call, and set the other party as the receiver of the call.
- **Start conference call-** The receiver can share with other users connected on the same network the received data from the streamer. After setting the call, the receiver can start a conference call by initiating a conference call. The participants will be selected via phone-numbers, and the receiver will send the data received to

other users in the conference. It is necessary that the other participations will be on the same WIFI network of the receiver – the video and audio streams **will not be available** to 3G network users and users in other WIFI networks. The users should approve their participation in the conference call (by using the ***Accept conference call*** functionality).

- **Stop capture video and vocal -** This functionality describes the situation in which the streamer stops capturing and sending video and vocal streams. At this point, the roles of "Streamer" and "Receiver" are no longer assigned to any participant of the call.

Receiver's Functionalities:

- **Accept a call-** Each user of the application who is connected to the phone's network can get an incoming call. This functionality allows the user to approve the incoming call and allows the connection between the parties (using the ***Session Manager***, shown in *figure 2*). The channel created between the parties is used for transmission of the video and vocal data, and also data used for managing the connection and the data transfer (by the ***Communicator***, as shown in *figure 2*).
- **Stop conference call-** the party who started the conference, can stop sharing the data between the other users.

Both-sides Functionalities:

- **Hang up a call-** Each party of the call can end it. This functionality closes the connection between the parties (by the ***Session Manager***, as shown in *figure 2*) and stops the video and vocal transmission, if transmitted.

Conference-receiver Functionalities:

- **Accept conference call-** This functionality describes the situation in which a user in the receiver's network accepts participation in a conference call, and can now starts receiving the video and vocal streams to his phone.
- **Stop participate in a conference call-** Each one of the passive users in the conference call can stop his participation in it, in a way which doesn't affect the rest of the call.

1.4.3 Output

- **Streamer's Outputs:**

  None.
- **Receiver's/Conference-Receiver's Output:**
  - **Video and Vocal –** The Video and Vocal captured by the Streamer's cell phone and received from the network.

1.5 – SYSTEM INTERFACES

1.5.1 Hardware Interfaces

Android Hardware Block Diagram

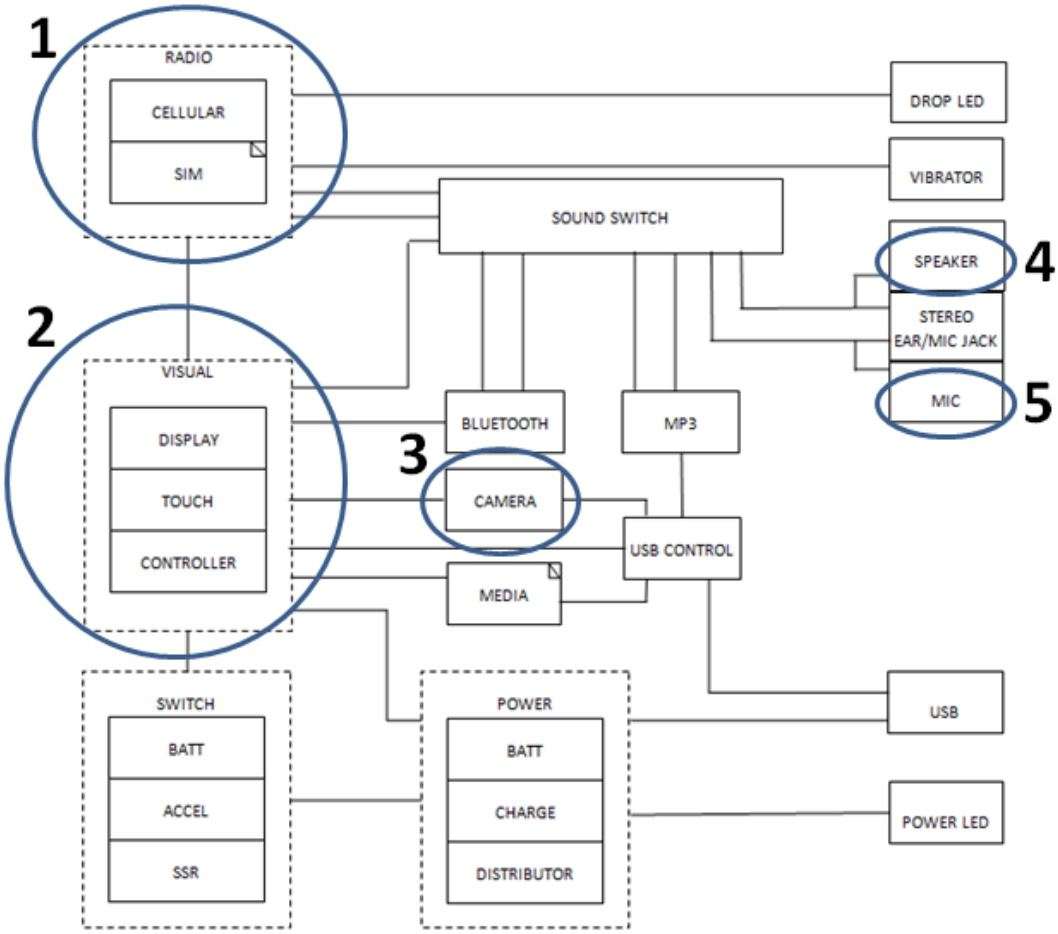


Figure 3 *– Android Hardware Block Diagram.*

1. **Radio (the Radio device in *figure 3*) –** Our software will interface with the Radio device of the android hardware. It will use it to send the SMS message at the begging of starting a call.
2. **Visual (the Visual device in *figure 3*) –** Our software will interface with the visual device of the android hardware. It will display the graphics of the program on the screen. In addition the user will communicate with the program using the touch screen and the controller device.

3. **Camera (the Camera device in *figure 3*) –** Our Software will interface with the camera device of the android hardware. It will connect to the camera and will be able to broadcast its video stream to the other party in the video call.
4. **Speaker (the speaker device in *figure 3*) –** Our software will interface with the speaker device of the android. It will play in the speaker, the vocal from the other party.
5. **Microphone (the microphone in *figure 3*)–** Our software will interface with the microphone device. It will be able to receive the vocal from the microphone and send it to the other party during the call.

Supported Devices

Our software will support all the cell phone with android operation system, version 1.6 and later.

Protocols

The network protocols interfaces that will be needed in our project are:

- **TCP –** This network protocol will be used for the connection between the two parties. This will create the connection link where the video stream and vocal stream will be sent.
- **UDP -** This network protocol will be used for transferring the video and vocal data streams between the two parties.
- **HTTP –** we will use the technique of *HTTP Tunneling*.
  HTTP Tunneling is a technique by which communications performed using various network protocols are encapsulated using the HTTP protocol. It allows the transfer between different networks without the knowledge of the user. We will use the technique in order to overcome firewalls and other obstacles that may reduce the transmitting rate.

1.5.2 Software Interfaces

Detailed Android Software Diagram

In this figure we see the components that assemble the major components in the android operating system.

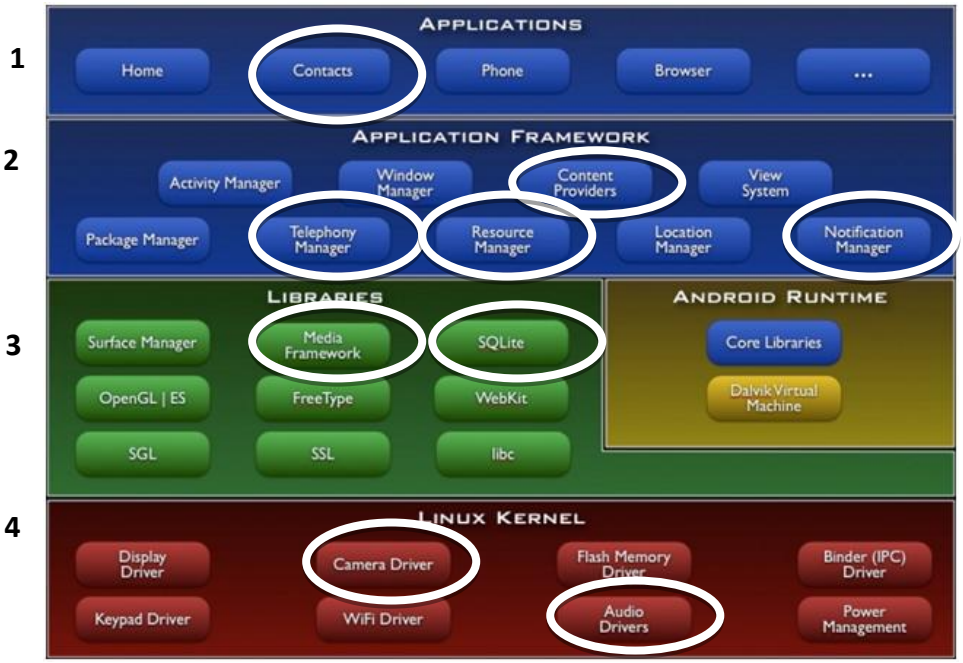


Figure 4 *–Detailed Android Hardware Block Diagram.*

We will now describe the interfaces between our project's software with each of the main layers in the Android software system:

1. **APPLICATIONS:**

   This layer contains the code of the applications installed on the android operation system. Our software will be added to this layer and will be a part of the installed application on the android software.

   The applications that will work with our software are:

- **Contacts –** Our project will supply an add-on for the Contacts application, installed on the phone. A user using the contacts application will be able to select a contact on its phone for calling through our application.

2. **APPLICATION FRAMEWORK:**

   The Layer contains all the toolkits on the android operating system. The toolkit helps to manage the phone basic functions. Some of the toolkits are provided by Google and parts are extensions or services

that the developer writes. The toolkits that will work with our software are:

- **Content Provider –** The Content Provider store **a**nd retrieve data and make it accessible to all applications. They are the only way to share data across applications because there is no common storage area that all android packages can access. In our application we will work with the content provider of the Contacts for retrieving contact persons for the use of starting a call.
- **The Telephony manager –** The Telephony Manager help to get information about the telephony services on the device. In our application we will work with the telephony manage for The manager detecting incoming telephone calls. When the application works and for sending the initial SMS message to the other party.
- **Resource Manager –** The Resource Manager helps to manage the resources of the application in example images and string in the layout. It helps to organize the resources for providing compatibility with different configurations. In our application we will manage different resources for the regular screen layout and for the landscape mode of the layout. In addition we will need the Resource Manager for supporting different languages.
- **Notifications Manager** - The Notification Manager manage the notification window which helps sending different alerts to the user. In our application we will use the Notification Manager to communicate with the user and alert him on different subjects such as: missing video calls, notification that that user id during a video call and etc.

3. **LIBRARIES:**

This layer contains different core libraries for the use of various components. The Libraries that will work with our software are:

- **Media Framework – The** Media Framework support playback and recording of many popular audio and video formats such as: MPEG4,

H264, MP3, and etc. In our application we will use the Media Framework for decoding and encoding the video and vocal streams.

- **SQLite –** The SQLite manage a relational database for every application. In our application we will use the data base for saving the records for missing calls, unanswered calls and network details for helping Analysis the quality of the network.

4. LINUX KERNEL:

Android is a Linux based operating system, this layer contains drivers that communicate with the hardware directly. The drivers that will work with our software are:

- **Camera Driver –** The Camera Driver is used to capture images, and retrieve frames for encoding for the video. The driver manage the actual camera hardware. In our application we will use the camera driver for capturing frames by the streamer.
- **Audio Driver -** The Audio Driver is used to record vocal from the microphone hardware. In our application we will use the audio driver for recording the vocal by the streamer.

1.5.3 Events

- **Incoming call**

  This event describes the situation in which a video call is accepted in a mobile phone of the other party.

- **Ending a call**

  This event describes the situation in which one of the parties hangs up the call.

- **Call waiting**

  This event describes the situation in which one of the parties gets another incoming call during the current one.

- **Incoming conference call**

  This event describes the situation in which a conference call is accepted from a receiver's mobile in the same network, and the receiver transmits the video stream from his mobile.

- **End conference call**

  This event describes the situation of ending a conference call by the receiver's party (which initiated the conference call).

- **Out of signal problem**

  This event describes the situation in which the signal is lost (e.g. when entering an elevator).

- **Traffic and congestion changes**

  This event describes the situation in which traffic and congestion affects the quality of the video and vocal streaming between the streamer and receiver. This event relates to the current state of the network connecting between the parties.

2. FUNCTIONAL REQUIERMENTS

Requirements from User's Point-Of-View

2.1 CAMERA CAPTURE FUNCTIONALITY

2.1.1 Start capture

This is the main functionality of a video call. It enables one of the parties to start transferring the video to the other party. This can be done remotely if the capture needs to be initiated in a distant mobile, or directly from the user's mobile to transmit a video to another distant user.

2.1.2 Stop capture

This option will allow a video capture that has been started to stop permanently, meaning that the camera stops working and showing on the screen. It could be done remotely or directly on the users mobile. The call itself shouldn't be stopped by this functionality.

2.1.3 Pause capture

This option allows a video capture that has been started to pause. The camera view will still show on the screen, but no video will be transmitted. While the camera is paused at the transmitting party, the other party will see the last frame of the video before it was paused.

2.1.4 Snapshot option

The application should support taking a snapshot from the video stream, and saving it to the cell-phone memory. This is useful when the user wants to take a picture of the current video for further use.

2.2 CALL FUNCTIONALITY

2.2.1 Start call

This functionality enables the communication between the sender party and the receiving party. By using it, the two cell phones will connect through the different networks and can start their video transmission.

2.2.2 End call

This functionality stops the communication between the communicating parties. It will terminate the connection between the two cell phones and return the application on both to its initial state.

2.2.3 Settings changing

The application will enable the user to change the settings of the application:

2.2.3.1 Automatic availability

This setting defines whether we need the user consent to enable an incoming video call or the incoming call is enabled automatically.

2.2.3.2 Video window size

The user can choose from 3 pre-defined different sizes of the video window.

2.2.3.3 The ringtone of an incoming video call

The user can choose the ringtone for an incoming video call from the existing ringtone list in his mobile phone. Silence and vibrate can also be chosen.

2.2.4 Call waiting

In case that the user is in a middle of a video call, and there is a call pending, the user is asked whether to stop the current video call and accept the new one. If the user chooses to ignore the incoming call – the incoming call is declined.

2.3 ENCODING FUNCTIONALITY

2.3.1 Encode video and voice streams

This functionality processes the video and audio streams received from the camera and microphone devices installed in the smart-phone. It will encode the streams using defined codecs for producing data that will pass through the network to the other parties of the conversation. This functionality will use codecs that maintain reasonable quality while producing small amount of data.

2.3.2 Decode video and voice streams

This functionality processes the video and audio streams received from the network connection. It will decode the streams using defined codecs for producing the video frames to display on the phone's screen, and the voice to play on the phone's speaker. This functionality will use the codecs defined earlier, according to the specific requirements.

3. NON FUNCTIONAL REQUIERMENTS

3.1 – PERFORMANCE CONSTRAINTS

3.1.1 – SPEED, CAPACITY & THROUGHPUT

- When the user starts a call, there should be a connection request in the callee cell phone in 5 seconds.
- There should be a delay of 3 second between the video broadcasting and the images in real life.
- There can be up to 20 users connected to the conference call.
- The program should return to its initial state in 2 seconds after ending a call.
- In case of disconnection from the local network that the phone is connected to, the system should alert the user in 5 second about the disconnection.

3.1.2 – RELIABILITY

Our system reliability will be affected by two main causes:

- Quality of the network which the users are connected through.
- The speed rate of the Android device processor.

3.1.3 – SAFETY AND SECURITY

Our system will not handle security issues because the safety and security is already implemented in our system by design. As mentioned above, the communication based on the P2P model, without any server as "middle-man". In this case, the only option for security failure is by eavesdropping to one of the cell phones, which is problematic – but not a part of the system.

3.1.4 – PORTABILITY

The application is designed for smart-phones running Android operating system, version 1.6 or later.

User interface will be written in English, with an option for extending to other languages in advanced stages of the development.

3.1.5 – OUR SYSTEM'S RELIABILITY

- If a delay of more than 5 seconds will occur between the streamer and the receiver – than the call will be disconnected.
- If less than half of the data packets transmitted by the streamer received on the receiver side - the system will sample the other network connection (3G or WIFI, according to the current network connection) for considering replacing the current connection type to the other. If no better condition was found in the other connection type, the call will be disconnected.
- One of every 1000 calling attempts will fail as a result of synchronization problems of the system.

3.1.6 – USABILITY

Our application has no Graphical User Interface (GUI), and is based on the GUI supplied by the Android OS and the *Contacts* application. Therefore, the usability of our application is derived from the usability of the Android OS and its supplements.

Our software does not delay or burden the base system, and therefore the usability is kept.

3.1.7 – AVAILABILITY

The software will be 100% available for the user. This is based on our system's design, which does not include any server connecting the two parties and acting as a middle-man. The only constraint is that the required user has the application and is connected to the network.

3.2 – PLATFORM CONSTRAINTS

- The development language of android is Java so our program will be developed in java with the Android SDK that is necessary for creating an android application.

3.3 – SE PROJECT CONSTRAINTS

- Our program is interactive and the inputs come from a real person – the user of the application.
- The device will not be simulated. It will be presented as is – working on real Android Phone.
- We will use samples of actual data in our simulation: real camera captures real WIFI network connection, etc.
- Our application requires two Android cell phones with a camera and a microphone. Any Android versions of 1.6 and later will satisfy the needs.
- We will start developing our program using the emulator program that will be installed on our computers. Later on we will test our program on two real Android cell phones in different networks.
- Presentation of the final project – we will demonstrate the application on the android cell phones and will show a video call between two users, through a real network.

3.4 – SPECIAL RESTRICATIONS AND LIMITATION

none.

4. USAGE SCENARIOS

4.1 - USER PROFILES – THE ACTORS

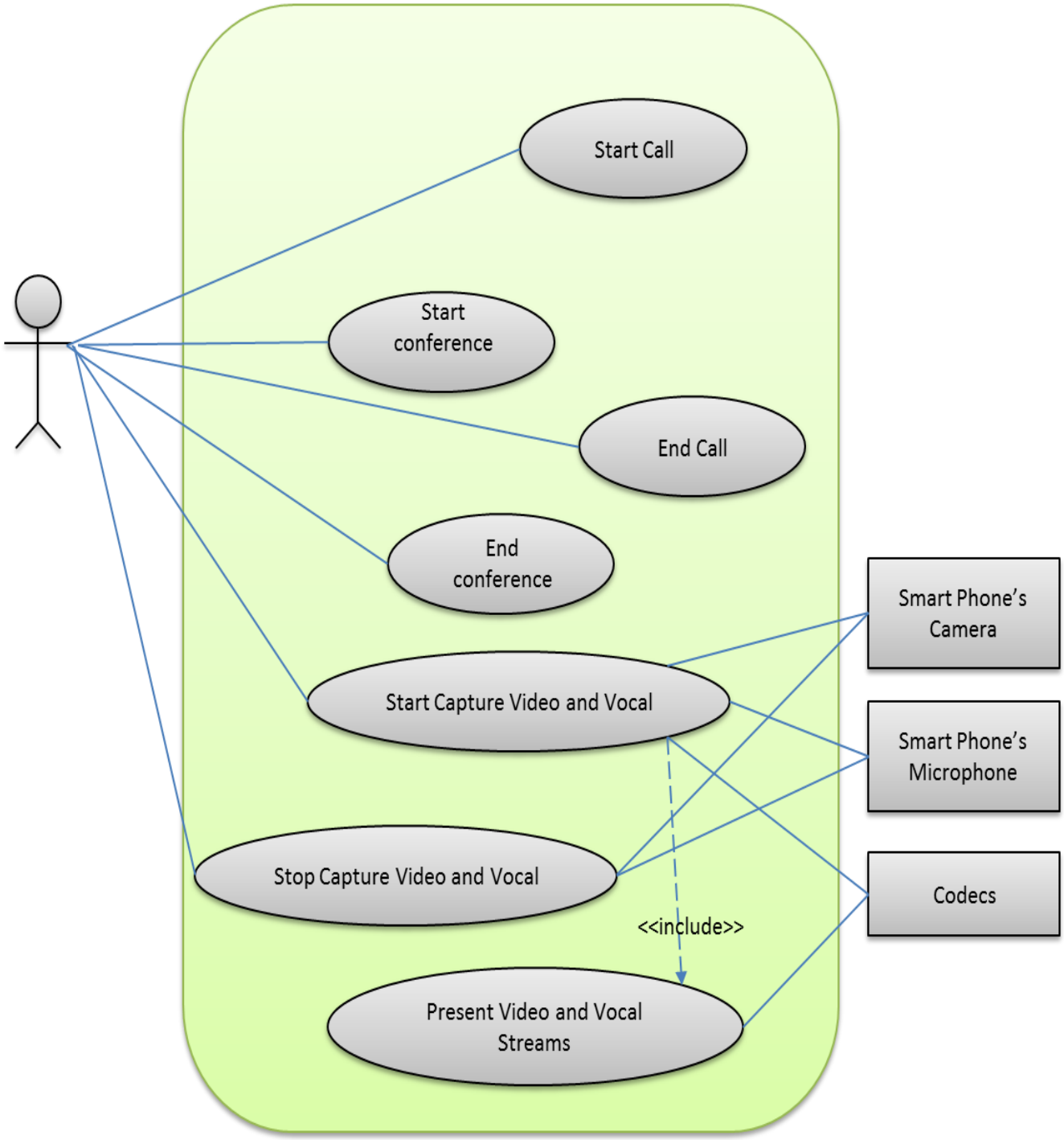
4.1.1 The Streamer Party

This actor uses the system to publish and transmit its video and voice inputs received from his device to the receiving party of the conversation. Its main use of the system is to encode its video and audio streams for transmitting through the network to the other party of the call. This actor also uses the system to receive calls from other actors of the system, and hang up the calls.

4.1.2 The Receiving Party

This actor uses the system to receive video and audio streams from the transmitting party of the conversation. Its main use of the system is to decode the video and audio streams received from the transmitting party through the network, for displaying and playing on the receiver device. This actor also uses the system to initiate calls to other available actors of the system, and hang up the calls.

4.2 – USE-CASES


Start Call
Start conference
End Call
End conference
Start Capture Video and Vocal
Stop Capture Video and Vocal
<<include>>
Present Video and Vocal Streams
Smart Phone's Camera
Smart Phone's Microphone
Codecs

4.2.1 UC1- Start a Call

Primary actor: User

Description: The user starts a call to contact a person from his contact list.

Stakeholders and interests: The user - wants a convenient and fast process to start a call to other users of the service.

Preconditions: The user must be connected to a network and run the application.

Post conditions: Connection has established between the two users, and each one of them can now become the streamer.

Main success scenario:

1. The initiator user (caller) selects the contact user that he wishes to call (callee).
2. The caller presses the "Call" Button.
3. The caller's application sends SMS message to the callee with the IP and port of the caller.
4. A service of the system on the callee side samples the inbox and recognizes the special message.
5. The system notifies the callee of an incoming call.
6. The callee approves the call, and the connection is established.

Alternative flows:

- At any time, system fails: the caller will be informed of a problem, and will be requested to try to use the service later.
- The requested contact is not connected to a network: the user will be informed that the contact person is un available, and that the connection can't be established.
- The contact disapproves the call: the user will be informed that the call has been rejected.

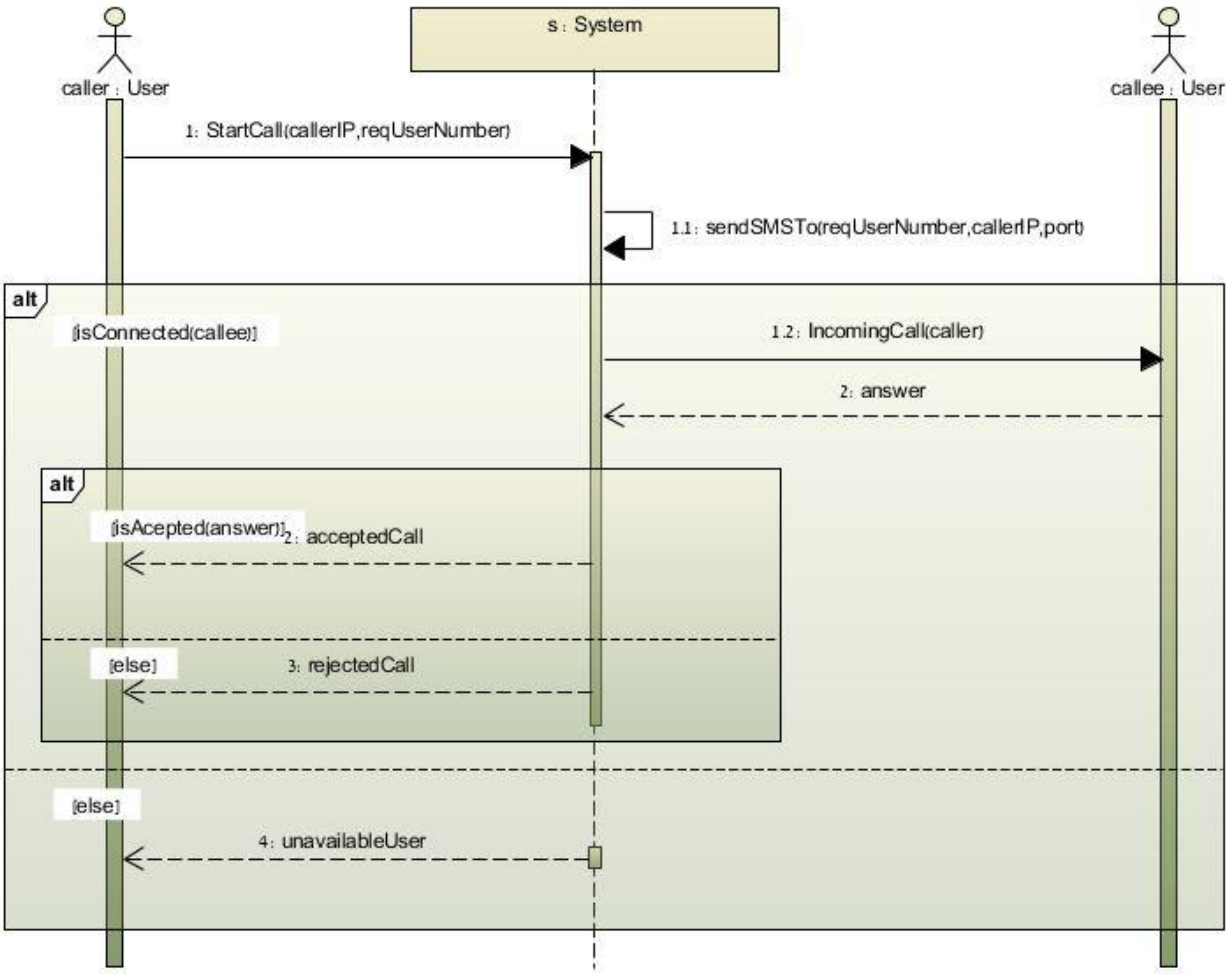


Figure 5 *–Sequence Diagram of Start a Call.*

4.2.2 UC2- End a Call

Primary actor: User

Description: The user ends a current call with the other party.

Stakeholders and interests: The user - wants a convenient and fast process to end a current call with other party.

Preconditions: The users must be during a call, and the video and audio capturing has stopped.

Post conditions: The connection between the two users is terminated.

Main success scenario:

1. The user presses the "End" button in his phone.
2. The system informs the other party of the end of the call, and disconnects the call between the two parties.
3. The application returns to its initial state.

Alternative flows:

- At any time, system fails: the user will be informed of a problem, and will be requested to try to use the service later.

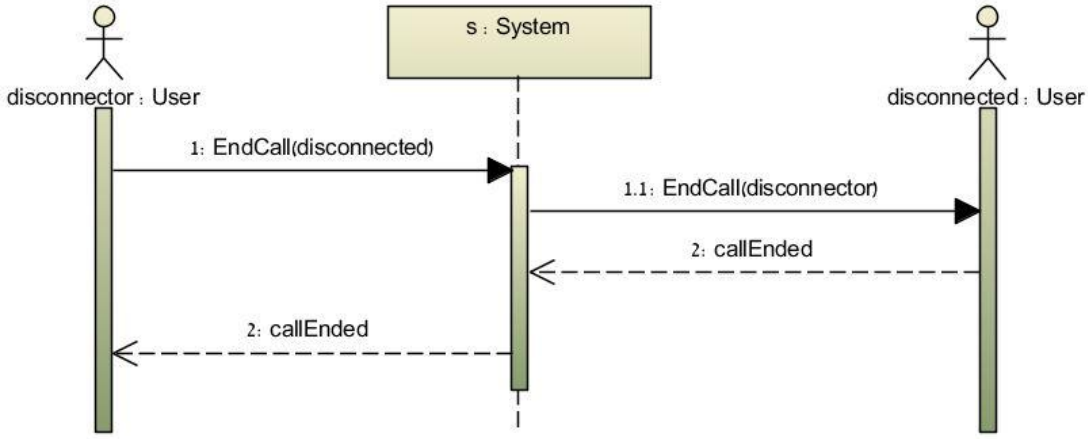


Figure 6 *–Sequence Diagram of End a Call.*

4.2.3 UC3- Start a Conference

Primary actor: receiver: User

Description: The receiver shares with other users connected on the same network the received data from the streamer.

Stakeholders and interests: The receiver - wants a convenient and fast process to spread the data received from the streamer to other selected users in the network.

Preconditions:

- The receiver must be during a call.
- The receiver and the users selected must be on the same WIFI network.

Post conditions: Connection has established between the receiver and the selected contacts, and the video and vocal streams are transmitted to them.

Main success scenario:

1. The receiver presses the "Conference" button in his phone.
2. The system opens the contact list for the receiver to choose the desired contacts for the conference call.
3. The receiver chooses the desired contacts.

4. The system sends for every requested user a conference call request.
5. The users approves the conference call, and the connection is established.
6. The receiver transmits to the requested users the data received in his phone.

Alternative flows:

- At any time, system fails: the user will be informed of a problem, and will be requested to try to use the service later.
- A requested contact is not connected to the same WIFI network: the user will be informed that the specified contact person is unavailable.
- A requested contact disapproves the call: the user will be informed that the call has been rejected by the specified contact person.

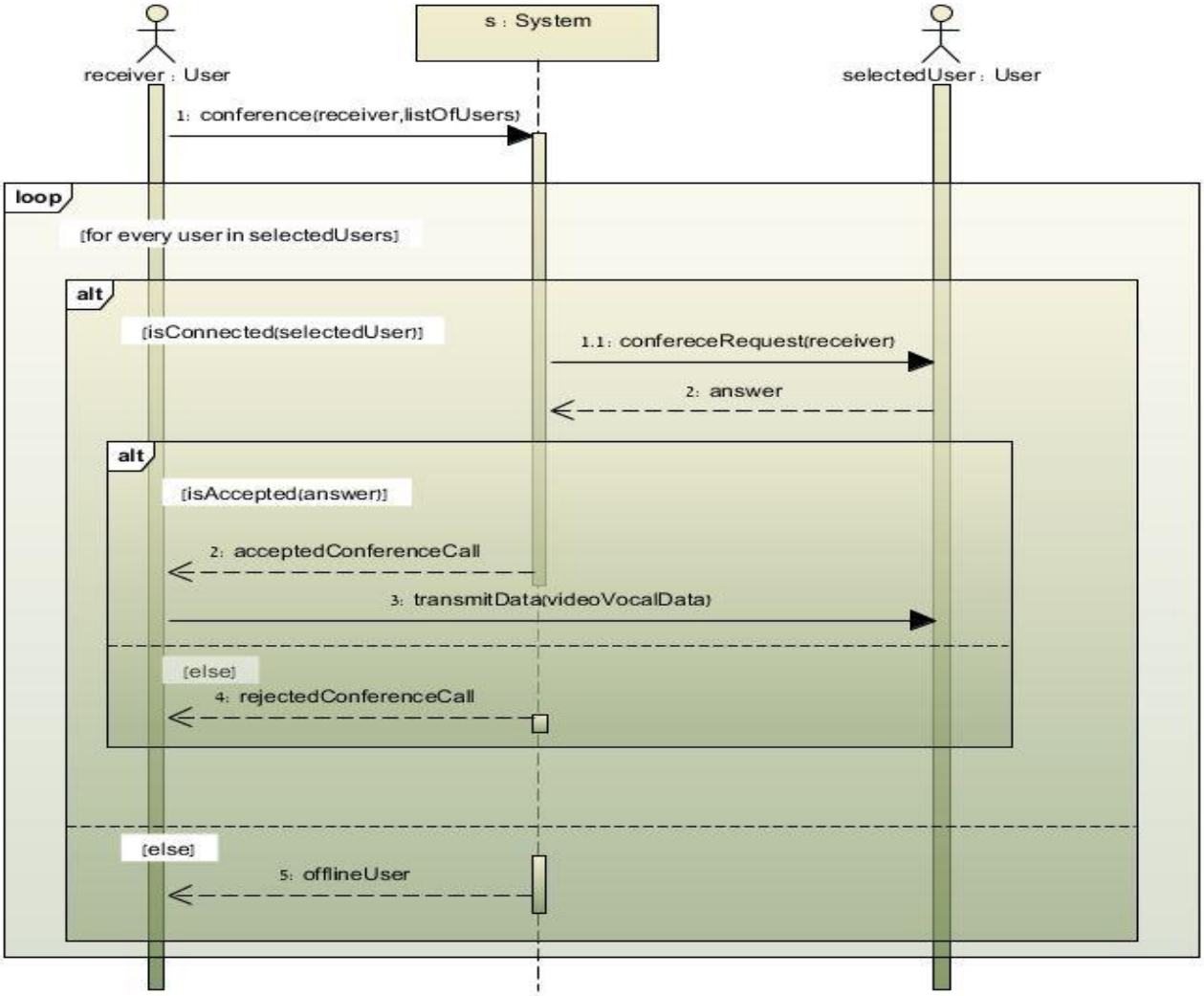


Figure 7 *–Sequence Diagram of Start a Conference.*

4.2.4 UC4- End a Conference

Primary actor: receiver: User

Description: The receiver who shares data with other users ends the conference call.

Stakeholders and interests: The receiver - wants a convenient and fast process to end the current conference call with the other users.

Preconditions: The receiver and the users must be during a conference call.

Post conditions: The connection between the receiver and the other users is terminated.

Main success scenario:

1. The receiver presses the "End Conference" button in his phone.
2. The system informs all the other users of the end of the call, and disconnects the call between the receiver to each one of them.
3. The system informs the receiver of the ending of the conference.
4. The application returns to its initial state.

Alternative flows:

- At any time, system fails: the user will be informed of a problem, and will be requested to try to use the service later.

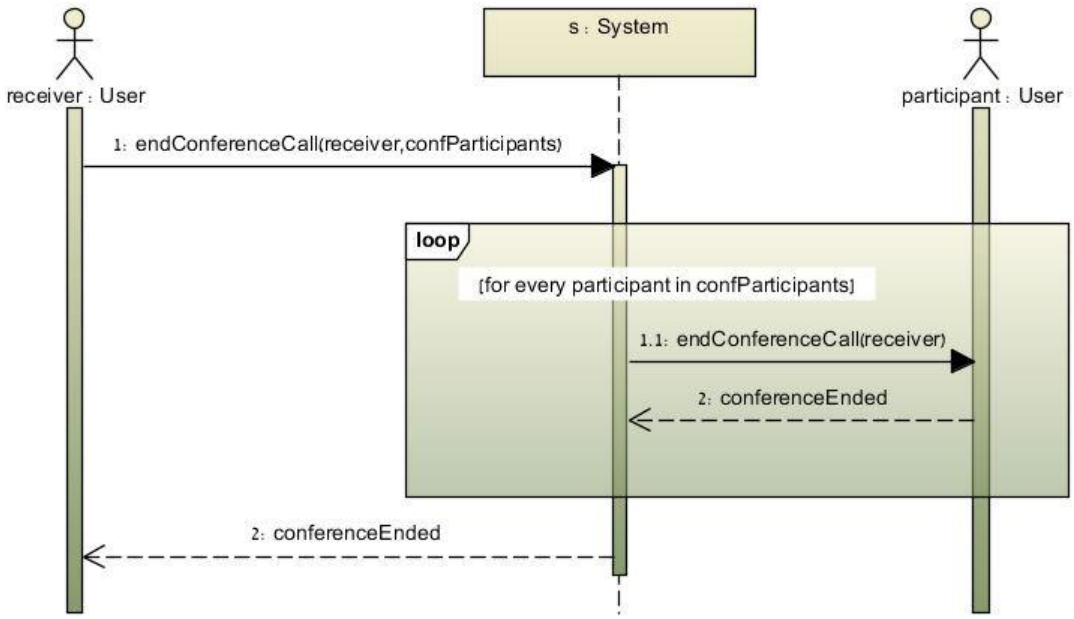


Figure 8 *–Sequence Diagram of End a Conference.*

4.2.5 UC5- Start capturing Video and Vocal

Primary actor: streamer: User

Description: The streamer starts capturing video and vocal streams from his camera and microphone.

Stakeholders and interests: The streamer - wants a convenient and fast process to start capturing video and vocal streams to the receiver party.

Preconditions: The streamer must be during a call.

Post conditions: The streamer's cell phone starts capturing video and vocal streams from the camera and microphone, and encoding it for transmitting to the receiver.

Main success scenario:

1. The streamer selects the "Start Capture" button in his phone.
2. The streamer's application starts the video and vocal capturing.
3. The streamer's application starts encode the data.
4. The streamed data is presented on the receiver's phone, via the use case *UC7 Present Video and Vocal Streams*.

Alternative flows:

- At any time, system fails: the user will be informed of a problem, and will be requested to try to use the service later.
- In case of problem in communicating with the smart phone's camera or microphone: the application will display an appropriate message.

In case of problem with encoding the captured data: the user will be informed of unsuccessful encoding process, and failed transmission of the captured data.

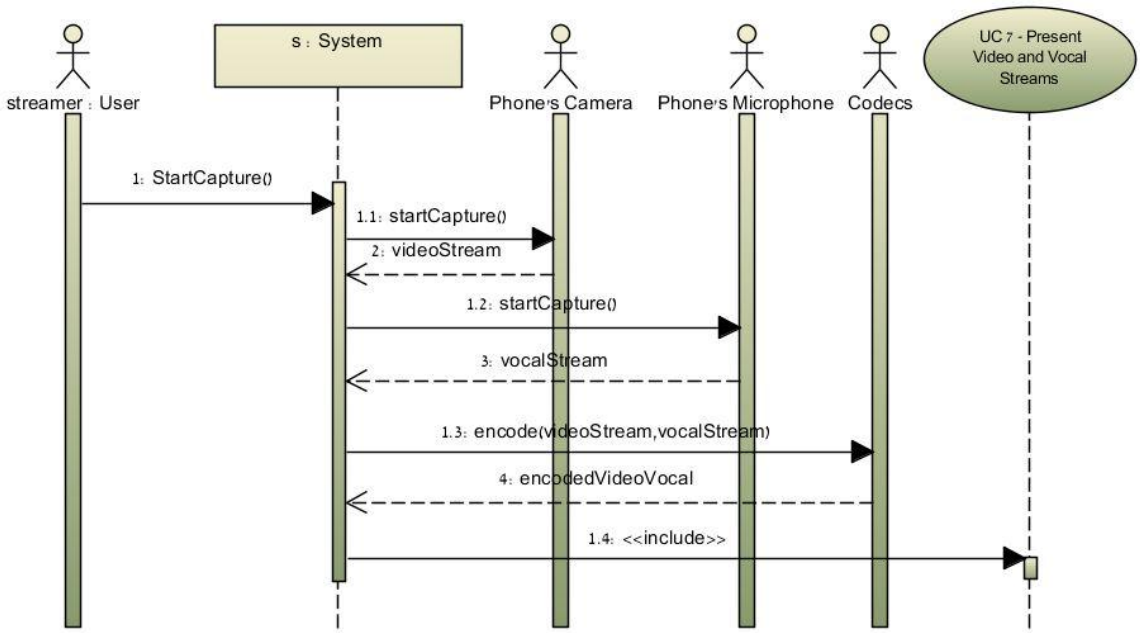


Figure 9 *–Sequence Diagram of Start Capturing Video and Vocal.*

4.2.6 UC6- Stop capturing Video and Vocal

Primary actor: streamer: User

Description: The streamer stops capturing video and vocal streams from his camera and microphone.

Stakeholders and interests: The user - wants a convenient and fast process to stop capturing video and vocal streams, to control what is visible to the other party.

Preconditions: The user must be during a call, and after starting the functionality of start capturing video and vocal.

Post conditions: The user's cell phone stops capturing video and vocal streams from his camera and microphone.

Main success scenario:

1. The user selects to stop capturing video and vocal streams.
2. The user's application stops the video and voice capturing.

Alternative flows:

- At any time, system fails: the user will be informed of a problem, and will be requested to try to use the service later.

- In case of problem in communicating with the smart phone's camera or microphone: the application will display an appropriate message.

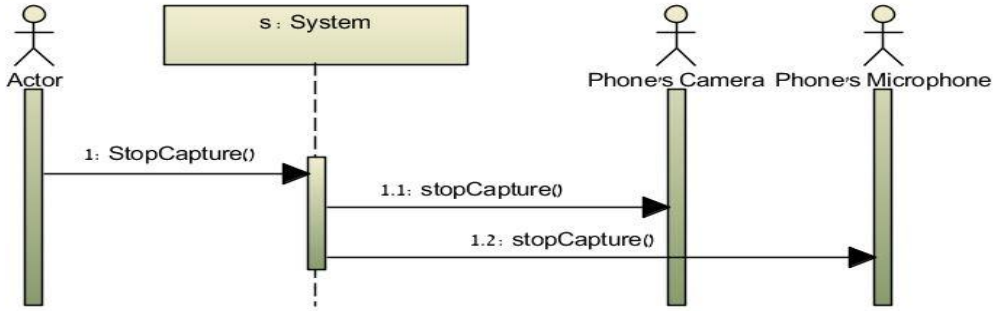


Figure 10 *–Sequence Diagram of Stop Capturing Video and Vocal.*

4.2.7 UC7- Present Video and Vocal Streams

Primary actor: None

Description: The receiver's phone displays video and vocal streams sent by the streamer.

Stakeholders and interests:

- The receiver - wants to be able to watch the video and hear the vocal captured by the streamer in a convenient layout.
- The streamer- wants to be able to transmit data to the receiver.

Preconditions:

- Both users must be during a call.
- The receiver has received the streams transmitted through the network, in order to present them.

Post conditions:

The receiver's phone presents the video vocal of the streamer party.

Main success scenario:

1. The user's system receives encoded video and audio streams.
2. The system decodes the received streams.
3. The system displays the decoded video and plays the audio to the user.

Alternative flows:

- At any time, system fails: the user will be informed of a problem, and will be requested to try to use the service later.
- The streamer has stopped transmitting video and audio streams: black screen will be displayed in the video window.
- In case of problem with decoding the received data: the user will be informed of unsuccessful decoding process if the data is large, otherwise it will try and proceed with the new data received.

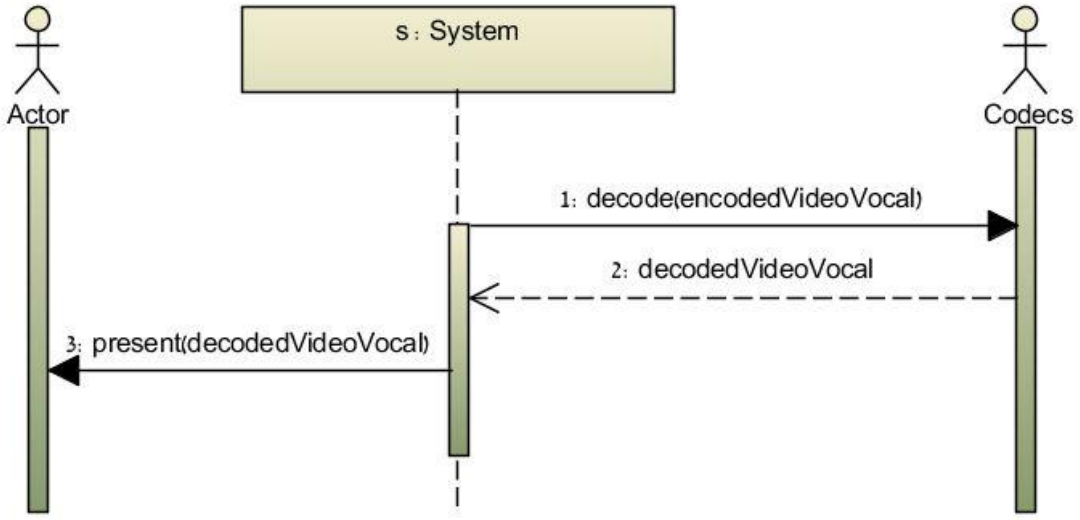


Figure 11 *–Sequence Diagram of Present Video and Vocal Streams.*

4.3 – SPECIAL USAGE CONSIDERATION

NA.

5. APPENDICES

5.1 - GLOSSARY

5.1.1 Android: Android is a mobile operating system initially developed by Android Inc., a firm purchased by Google in 2005. Android is based upon a modified version of the Linux kernel. Android has a large community of developers writing application programs *("apps")* that extend the functionality of the devices. Developers write in the Java language, controlling the device via Google-developed Java libraries. The Android software development kit (SDK) includes a comprehensive set of development tools. These include a debugger, libraries, a handset emulator (based on QEMU), documentation, sample code, and tutorials. Currently supported development platforms include computers running Linux (any modern desktop Linux distribution), Mac OS X 10.4.9 or later, Windows XP or later. The officially supported integrated development environment (IDE) is Eclipse (currently 3.4 or 3.5) using the Android Development Tools (ADT) Plugin, though developers may use any text editor to edit Java and XML files then use command line tools (Java Development Kit and Apache Ant are required) to create, build and debug Android applications as well as control attached Android devices (e.g., triggering a reboot, installing software package(s) remotely).

5.1.2 Codecs: A codec is a device or computer program capable of encoding and/or decoding a digital data stream or signal. A codec encodes a data stream or signal for transmission, storage or encryption, or decodes it for playback or editing. Codecs are used in videoconferencing, streaming media and video editing applications. A video camera's analog-to-digital converter (ADC) converts its analog signals into digital signals, which are then passed through a video compressor for digital transmission or storage. A receiving device then runs the signal through a video decompressor, then a digital-to-analog converter (DAC) for analog display.

5.1.2.1 H.264/MPEG-4 AVC: H.264/MPEG-4 Part 10 or AVC (Advanced Video Coding) is a standard for video compression. H.264/MPEG-4 AVC is a block-oriented motion-compensation-based codec standard. H.264 is

used in such applications as players for Blu-ray Discs, videos from YouTube and the iTunes Store, web software such as the Adobe Flash Player and Microsoft Silverlight, broadcast services for DVB and SBTVD, direct-broadcast satellite television services, cable television services, and real-time videoconferencing. The intent of the H.264/AVC project was to create a standard capable of providing good video quality at substantially lower bit rates than previous standards (e.g. half or less the bit rate of MPEG-2, H.263, or MPEG-4 Part 2), without increasing the complexity of design so much that it would beimpractical or excessively expensive to implement. An additional goal was to provide enough flexibility to allow the standard to be applied to a wide variety of applications on a wide variety of networks and systems, including low and high bit rates, low and high resolution video, broadcast, DVD storage, RTP/IP packet networks, and ITU-T multimedia telephony systems.

5.2 – SIMILAR PRODUCTS

5.2.1 Facetime: FaceTime is a video calling software developed by Apple for iPhone 4, the fourth generation iPod Touch, and computers running Mac OS X. FaceTime works by connecting an iPhone 4, 4th Generation iPod touch or computer with Mac OS X to another similar device. on the iPhone 3G and iPhone 3GS, users can enter facetime:// urls in Safari to trigger what appears to be a FaceTime call, although it does not truly ever successfully initiate. FaceTime is not compatible with non-Apple devices or any other video calling services. On the iPhone, FaceTime works in the Phone application instead of being a separate app. It can be activated when in the Phone application by placing a call, and pressing the FaceTime button. Although Apple limits FaceTime to Wi-Fi only, FaceTime over 3G can be achieved by using third party jailbreaking commercial software such as the FaceBreak, 3G Unrestrictor or My3G which fools the phone into believing that it is connected via Wi-Fi. These programs are **only available on jailbroken iPhones** via the Cydia St ore. At the moment **there are no known free of cost solutions** to unrestrict FaceTime. FaceTime uses about three megabytes of data per minute of conversation, so users with a limited data plan who use FaceTime over 3G (only on

jailbroken devices as of right now) must be careful to not overextend their data limit.

5.2.1 Qik: **Qik** is a mobile live video streaming and two-way video conferencing application that allows users to stream live video from their cell phones to the internet. Qik enables users to record and upload video directly from supported cell phones. Qik videos can be shared via Facebook, Twitter, Youtube, and many other social network sites. Qik was the first mobile video service to support Facebook Connect. It allows Qik users to automatically post videos directly to their video collection or wall and does not require them to install additional Facebook applications. Qik announced the first ever Android live streaming client in June 4, 2009 with the launch of the HTC Evo 4G. It is currently available from the Android Market.