Video conference Android platform by your mobile phone

Mr. Mohamed Khalifa
WIMCS Research Team, ENET'COM
Sfax-University, Tunisia

Dr. Chaafa Hamrouni
Telecommunication department
CERT
Sfax, Tunisia

Pr. Mahmoud Abdellaoui
WIMCS Research Team, ENET'COM
Sfax-University, Tunisia

***Abstract*— Video conferencing is a visual and audio communications technology dedicated to Smartphones. It is based on the client-server communication model that this requires many limits since using the server. In this paper, we proposed a new communication model without going through the server (client-client), in addition, this model makes video conference used by mobile multimedia phones.**

Keywords—videoconference; multimedia mobile phone; streaming; Smartphone; Server; Customers.

I. Introduction

On 1983, the first classic mobile phone is launched by Motorola; it transmits and receives text messages and calls. It often has a rudimentary camera and may be a game or two. After he was replaced by the media mobile phone that has many additional functions, made possible through the integration of an advanced operating system in the phone. The use of these two types of mobile phone gradually decreased to 1992. Since 1994, new cells phone categories are offered to the public: Smartphone that supported many functions of a PDA and a laptop. The use of these handheld devices are exploded in the world of mobile telephony, resulting in partial loss of other types of mobile phones even if they are expensive, subject to theft, are not within the reach of all world, easily breakable... In addition, a comparison was made between the mid-phones and multimedia mobile phones with the operating system level, memory, video and audio format, display protocol... The results of this comparison show that differences and characteristics between these two categories of cell phones are a little more subtle, and then they allow us to make the competitive multimedia cell phones to other ordis-phones. Similarly, these simple lightweight cell phones are intermediate between the Smartphone mobile and conventional phones. They have simplified and various multimedia features, a very good battery life, ease of use, much less fragile than the Smartphone and especially that mobile search is within the reach of everyone. But, they are less popular compared to Smartphones because of these applications. Faced with this gap between the two in uses and applications, you have to technically modify these multimedia mobile phones with the goal to adapt them and bring them level with Smartphone. For thus, in this article we created a usable application in the field of Streaming (video conferencing, broadcasting, remote monitoring, remote training ...) and make it adaptable to multimedia mobile phones. This application allowed transmitting video and audio data between two or more customers without using server. The created application, based on a new model audio and video communication without server, presented a good solution to resolve the weaknesses environment client-server such high cost, limited support, a weak link...

II. Theoretical Analysis

In this part, we have presented, at first, a video-conference constraints and secondly, the operation of this application with and without server.

A. Videoconference constraints

At a meeting in the company headquarters, the presence of all heads of agencies, which are scattered throughout the country, present difficulties; for that, they are obliged to perform a remote meeting through video conferencing. The latter is a widely used technology that allows two (or more) persons to enter remote visual and audio communication and work together (or organizing business meetings or conferences or distance learning courses). So, videoconferencing is the improvement of communications source of increased productivity, reduced costs, time lost from deletion, environmental conservation…. This technique requires a rate sufficient to provide transportation of the picture and sound and also three basic materials such as: microcomputer to display the picture, webcam or camera to capture the picture and microphone for sound. Most video-conferencing applications using a server, is high cost. For that, main goal of this article is the establishment of a videoconference Android platform across your multimedia mobile phone to achieve a "**client-client**" communication model without server.

B. Application with using server

There are two types of Streaming server: one standard server and the other specialized server. In this application, we used the dedicated server since it supports RTSP protocol, against the standard server supports HTTP. Similarly, most multimedia mobile phones support RTSP / RTP. For thus, we chose the RTSP protocol to control the delivery of video and audio data in real time with properties and RTP to transmit multimedia data in real time [1], [2]. The used server in this application is: "Wowza Media Server" [3].

RTP works always with its companion RTCP to allow monitoring of data delivery in a manner scalable to large multicast networks. UDP, that's the best underlying protocol, used for these two protocols as most multimedia mobile phones support RTSP / RTP non-interlaced (RTP over UDP) [4]. To transmit video and audio data in the network, it's best to encode the data with a low bit rate and low complexity encoding. For that, we use the H.263 encoder for the video data and the encoder AMRNB for audio data as most multimedia mobile phones support these encoders. After the selection of appropriate protocols and configuring encoding settings that are adaptable to the capabilities of multimedia mobile phones, we started to have how to build video and audio data and send them into the network. The construction of multimedia data is done in two classes, one for video and one for audio data using the same object "**MediaRecorder**" which is belongs to the super class "**Video_Audio_Data**".

```
mMediaRecorder = new MediaRecorder();
mMediaRecorder.setCamera(camera);
mMediaRecorder.setVideoSource(MediaRecorder.VideoSource.
CAMERA);
mMediaRecorder.setOutputFormat(MediaRecorder.OutputFormat.
THREE_GPP);
mMediaRecorder.setVideoEncoder(MediaRecorder.VideoEncoder.
H263);
mMediaRecorder.setPreviewDisplay(mSurfaceView.getHolder().
getSurface());
mMediaRecorder.setVideoSize(176,144);
mMediaRecorder.setVideoFrameRate(15);
mMediaRecorder.setVideoEncodingBitRate((int)(170*0.8));
mMediaRecorder.setOutputFile((LocalSocket)emetteur
.getFileDescriptor());
mMediaRecorder.prepare();     mMediaRecorder.start();
```

Fig. 1. Video data conception program.

```
mMediaRecorder = new MediaRecorder();
mMediaRecorder.setAudioSourc(MediaRecorder.AudioSource.CAMCORD
ER);
mMediaRecorder.setOutputFormat(MediaRecorder.OutputFormat.AMR_NB
);
mMediaRecorder.setAudioEncoder(MediaRecorder.AudioEncoder.AMR_N
B);
mMediaRecorder.setAudioChannels(2);
mMediaRecorder.setAudioSamplingRate(8000);
mMediaRecorder.setAudioEncodingBitRate(16000);
mMediaRecorder.setOutputFile((LocalSocket)emetteur.getFileDescript());
mMediaRecorder.prepare();  mMediaRecorder.start();
```

Fig. 2. Audio data conception program.

After the multimedia data establishment, we transferred these to the "**RTP_RTCP_Paquet**" class that can fill data camera and microphone in a RTP packet from the length of the header of the packet to the value MTU in the format of the header of RTP packet payload to use for AMRNB (audio) and H263 (video) using the class "**Data_AMRNB**" and class "**Data_H263**" in order to create an RTP packet that will be sent in unicast or multicast and an RTCP packet that will be sent every three seconds [5], [6].

```
RTP_RTCP_Paquet.setInputStream((LocalSocket)recepteur.
getInputStream());
RTP_RTCP_Paquet.start();
```

Fig. 3. Creat packet

Before starting the dissemination of multimedia data in real time, RTSP streams are exchanged between client-server and server-client to provide customers a URL can read the media on the server. These exchange messages are described by the following figure:

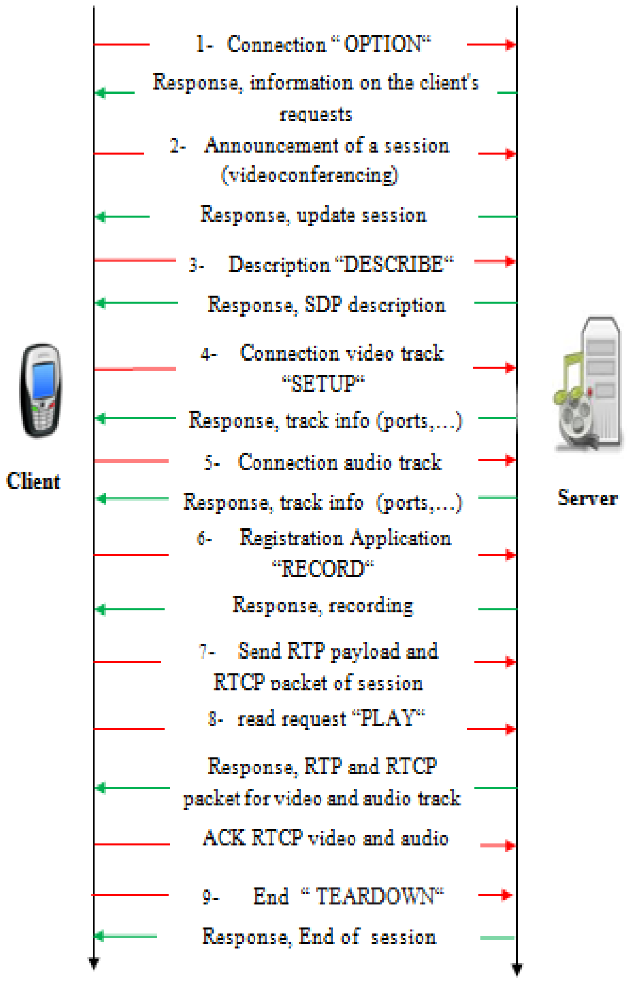
1- Connection " OPTION"
Response, information on the client's requests
2- Announcement of a session (videoconferencing)
Response, update session
3- Description "DESCRIBE"
Response, SDP description
4- Connection video track "SETUP"
Response, track info (ports,...)
5- Connection audio track
Response, track info (ports,...)
6- Registration Application "RECORD"
Response, recording
7- Send RTP payload and RTCP packet of session
8- read request "PLAY"
Response, RTP and RTCP packet for video and audio track
ACK RTCP video and audio
9- End " TEARDOWN"
Response, End of session
Client
Server

Fig. 4. Diagram of exchanges between the server and the client.

1. The client initiates the session with the server by sending a request "OPTIONS". The server responds to this request with information on what to support and what kind of requests it can receive.
2. After launching session, the client announces the session type: video conferencing, television ... The server responds with the session update.

3. The client sends a message describing. The server responds with an SDP file that the client can use to get more information about the content that will be sent by the server [7].
4. The client sends a setup request to the server to the video track at the end to inform the server that it will use UDP ports for RTP and RTCP communication. The server responds with information about the UDP ports that will be used by the server for this session.
5. The client sends a setup request to the server to the video track, the end to inform the server that it will use UDP ports for RTP and RTCP communication. The server responds with information about the UDP ports that will be used by the server for this session.
6. After the establishment of the communication ports, the client requests to record the session. The server saves the session to become accessible by other customers.
7. After the recording session, the client begins sending the RTP stream of the session.
8. The client sends a read request that informs the server that it is ready to receive the RTP data stream. The server starts sending the RTP payload and RTCP packet, also, the client sends RTCP packet to the server.

Here is a sample source code which describes how to read media data located on the server with the subject "MediaPlayer":

```
player.setAudioStreamType(AudioManager.STREAM_MUSIC);
player.setDataSource("rtsp://@IP_de_serveur:port_serveur/live/
test.stream");
player.setOnErrorListener(this);
player.setOnPreparedListener(this);
player.prepareAsync();
player.start();
```

Fig. 5. Read media data.

9. If the client closes the read a "TEARDOWN" request is sent to the server so it stops the streaming session.

C. *Application without using server*

In this part, we used the same encoders for video and audio data, but we only used the RTP and UDP transport protocols. In addition, we have optimized the messages exchanged between the client and the server to allow a multimedia cell phone to play the same role server to perform a visual and audio communication between two (or more) multimedia mobile phones without going through the server. This communication is based on the SDP. Hence session description record in the internal memory of the mobile device as a text file and read this file directly using a media player (VLC, FFmpeg, ...). These optimized exchanges messages are described by the following figure:

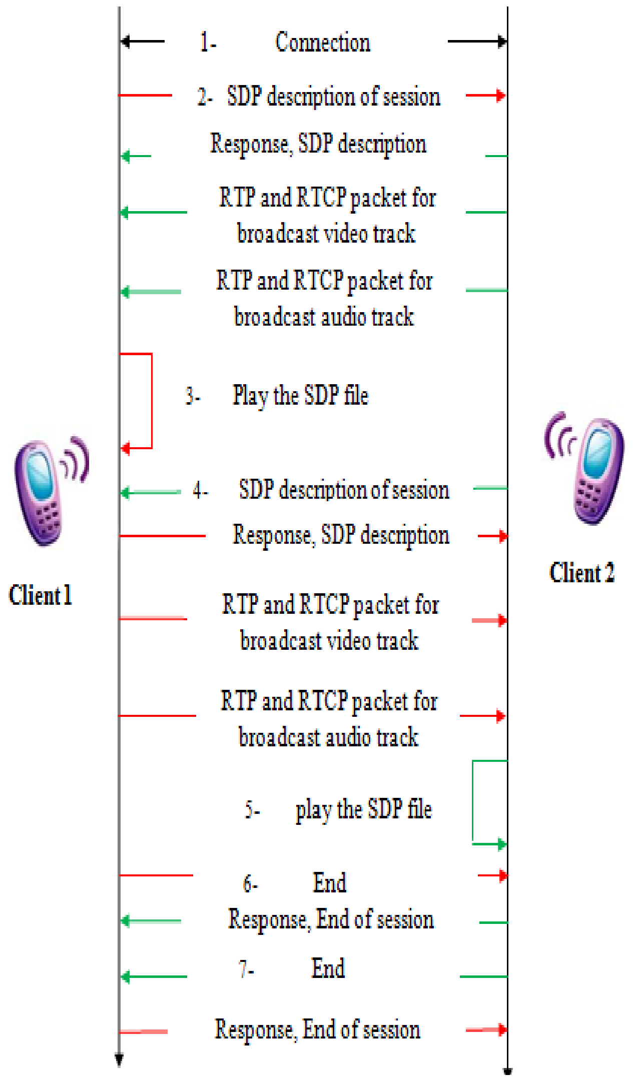
1- Connection
2- SDP description of session
Response, SDP description
RTP and RTCP packet for broadcast video track
RTP and RTCP packet for broadcast audio track
3- Play the SDP file
4- SDP description of session
Response, SDP description
RTP and RTCP packet for broadcast video track
RTP and RTCP packet for broadcast audio track
5- play the SDP file
6- End
Response, End of session
7- End
Response, End of session
Client 1
Client 2

Fig. 6. Diagram of exchanges between clients without server.

Before starting the session, the connection establishment between the two clients is done reliably using TCP / IP to transmit the SDP description. For cons, the RTP payload transmission and the report of the sender is unreliably using RTP over UDP. This is explained by the following steps:

1. Establishing connection.
2. After the connection establishment, the client 1 demands SDP description of the client 2. The latter responds with an SDP text file. The client 1 will also receive the RTP payload and RTCP packet.
3. The client 1 stores the SDP description in an internal memory as a text-file and then read that file.

```
v=0
o=- 0 0 IN IP4 null
s=Unnamed
i=N/A
c=IN IP4 192.168.168.103
t=0 0
a=recvonly
m=audio 5004 RTP/AVP 96
a=rtpmap:96 mpeg4-generic/16000
a=fmtp:96 streamtype=5; profile-level-id=15;mode=AAC-hbr;
config=1408; SizeLength=13; IndexLength=3; IndexDeltaLength=3;
a=control:trackID=0
m=video 5006 RTP/AVP 96
a=rtpmap:96 H263-1998/90000
a=control:trackID=1
```

Fig. 7. Description of SDP

4. Similarly, the client 2 requests the SDP description. The client 1 responds with a text file also it will send the RTP payload and RTCP packet.
5. The client 2 stores the SDP description in an internal memory as a text-file and then read that file.
6. If the client 1 closes the playback. The client 2 also closes the streaming session.
7. If the client 2 closes the playback. The client 1 also closes the streaming session.

This application is not limited only between devices but it can be used by a group of multimedia mobile phones. Hence, every customer of the group sends the session description to a group of receivers. This group is specified by a multicast IP address (224.0.0.0 to 239.255.255.255) [8].

III. Study of Cases

In this part, we studied two cases of this application, one with server and another without server.

A. *Case study with using server*

This application allows a user to open two sessions (video session and audio session):

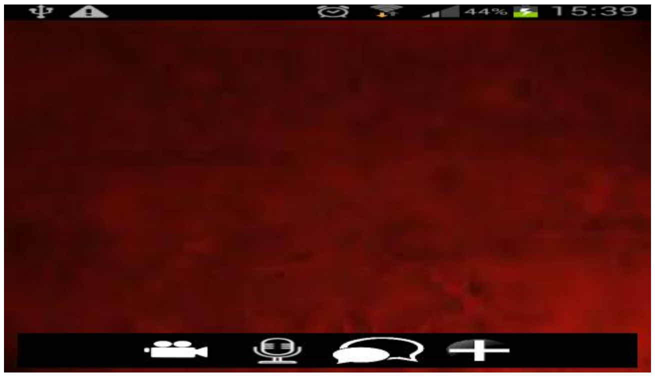

Fig. 8. Home application.

If a user wants to open a video session he has to press the image "camera". Hence, the video session activity will be created:

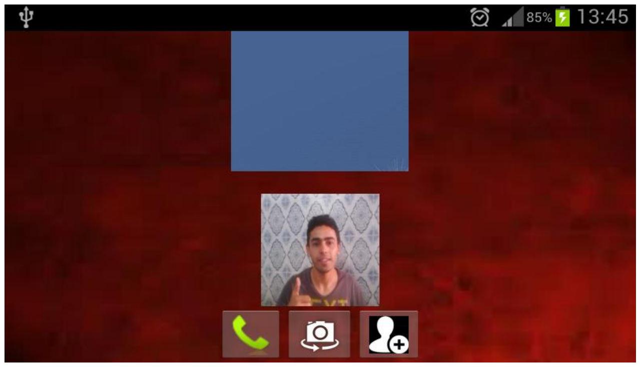

Fig. 9. Activity video session.

If the user wants to establish a connection with the server, he must press the "call" image. So, a dialog box will be displayed to enter the authentication settings for the server and select the video quality settings:

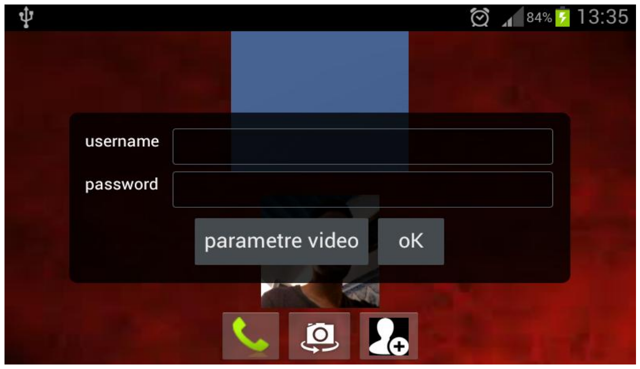

Fig. 10. Box authentification.

To connect to the server, the user must press the "OK" button. After the connection establishment, the server saves the session of the client and it will also receive data from camera and microphone for this session. If the user wants to open a visual and audio communication, it must press the "pseudo" image to enter the nickname of the other participant:

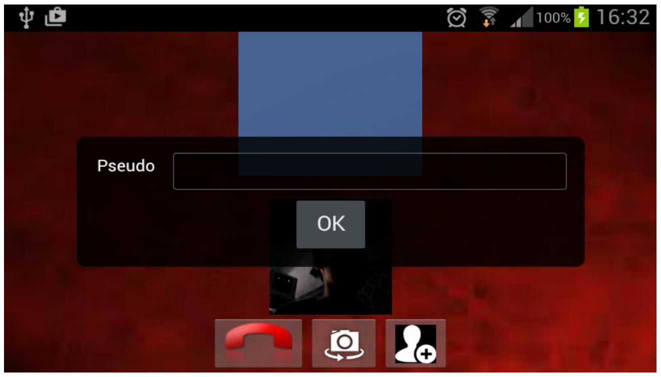

Fig. 11. Pseudo.

After adding nickname, the user must press the "OK" button to perform a visual and audio communication:

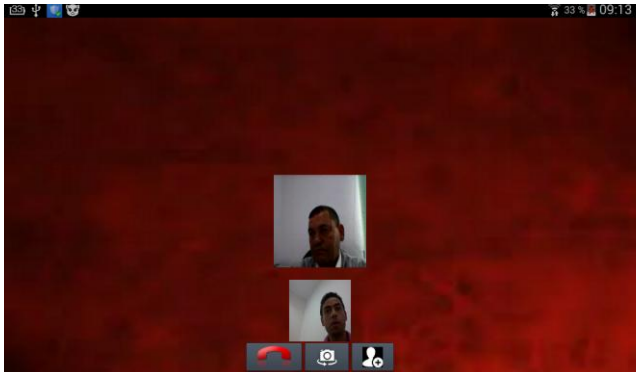
Fig. 12. Transmitter side.

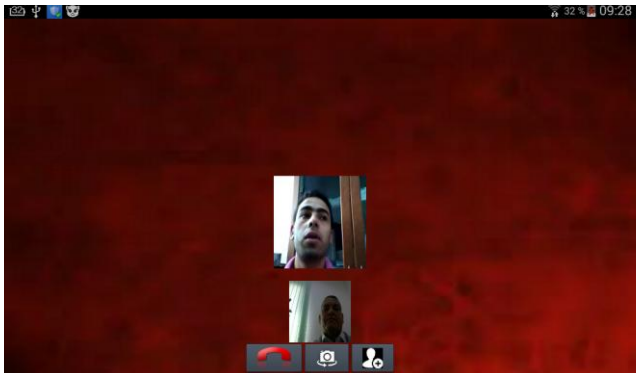
Fig. 13. Receiver side.

B. Case study without using server

After creating the activity video session, the user must press on the image "call" to enter the identifiers of the other Participant:

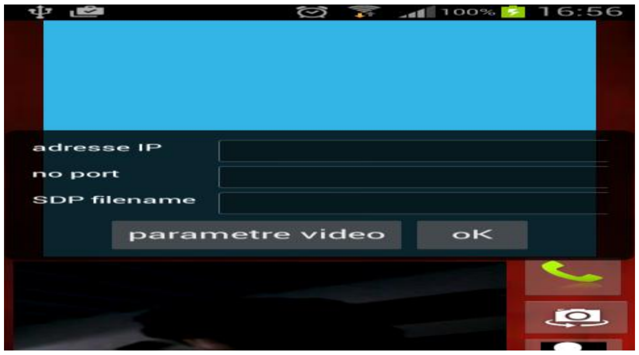

Fig. 14. Connection parameter.

After the connection establishment, a text file containing the SDP description will be stored in the internal memory of each client. For now, in this application reading the file is through a media player like VLC to make a visual and audible communication:

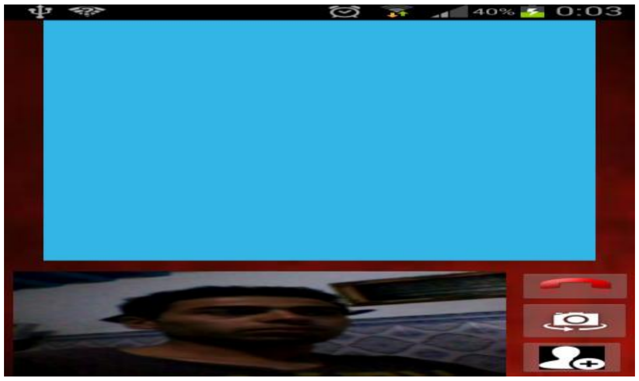
Fig. 15. Transmitter side.

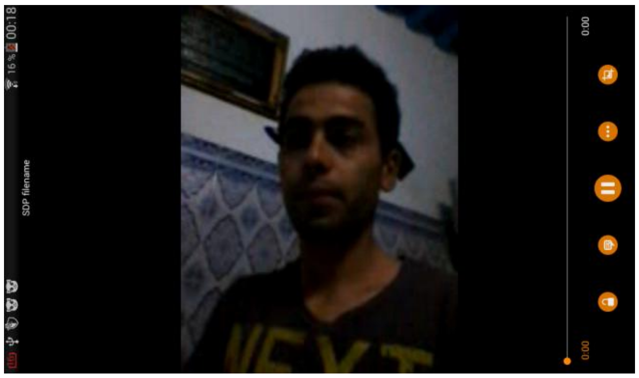
Fig. 16. Read SDP file with VLC.

IV. Conclusion

In this work, we have established a new model of video and audio communication from multimedia mobile phone without going through the server. This model is more efficient than the existing model at the level of simplicity, cost, use... The advantages offered by the implementation of this model are manifested in adaptation capabilities of all mobiles that contain three basic material video conferences (computer, camera and microphone).

Abbreviations and Acronyms

PDA: Personal Digital Assistant; RTSP: Real Time Streaming Protocol; HTTP: Hyper Text Transfer Protocol; RTP: Real-time Transport Protocol; RTCP: Real-time Transport Control Protocol; UDP: User Datagram Protocol; H.263: Video Codec; AMRNB: Adaptive Multi-Rate Narrow-Band; MTU: Maximum Transfer Unit; URL: Uniform Resource Locator; SDP: Session Description Protocol; TCP: Transmission Control Protocol; IP: Internet Protocol.

ACKNOWLEDGMENT

This work has been accomplished at WIMCS-Team research, ENET'COM-Sfax University. Part of this work has been supported by CERT-Tunisia Telecom and MESRSTICS Scientific Research Group-Tunisia.

REFERENCES

[1] R. Lanphier and H. Schulzrinne, "Real Time Streaming Protocol", IETF, REF 2326, Avril 1998, pp. 92.

[2] H. Schulzrinne, S. Casner, R. Frederick and V. Jacobson "RTP: A Transport Protocol for Real-Time Applications", Audio-Video Transport Working Group, REF 1889, January 1996, pp. 75.

[3] The Wowza Media Server. Available http://www.wowza.com/pricing/installer.

[4] J. Postel, "User Datagram Protocol", IETF, REF 768, 28 august 1980, pp. 3.

[5] J. Sjoberg, M. Westerlund, A. Lakaniemi and Q. Xie, "Real-Time Transport Protocol (RTP) Payload Format and File Storage Format for the Adaptive Multi-Rate (AMRNB) and Adaptive Multi-Rate Wideband (AMRWB) Audio Codec, IETF, REF 3267, june 2002, pp. 49.

[6] J. Ott, C. Bormann, G. Sullivan, S. Wenger and R. Even, "RTP Payload Format for ITU-T Rec. H.263 Video", IETF, REF 4629, January 2007, pp. 29.

[7] M. Handley, V. Jacobson, "SDP: Session Description Protocol", IETF, REF 2327, April 1998, pp. 42.

[8] S. Deering and Stanford University, "Host Extensions For IP Multicasting", IETF, REF 1054, May 1988, pp. 19.