SIMULATORS OF VOIP TRAFFIC IN P2P NETWORKS

LEONARDO PASINI AND SANDRO FELIZIANI

Departments of Mathematics and Computer Science,
University of Camerino,
Via Madonna delle Carceri, 62032 Camerino, Italy
leonardo.pasini@unicam.it

(Received 25 March 2009; revised manuscript received 10 May 2009)

Abstract: This study is included in a research programme developed by the authors at the University of Studies in Camerino for the construction of traffic flow simulators in computer and telecommunication networks [1–4]. The article contains the definition of a library of object types whose architecture is based on the queue networks, for a simulation of the VOIP traffic in P2P networks. Basing on this library, we simulate the traffic in a P2P network that is locally implemented with three different technology types:

1. Fast Ethernet,
2. Wireless,
3. ADSL2+.

In this context, we simulate traffic flows due to file-sharing applications in the network and due to voice communications through the VOIP technology. Thus, we evaluate the impact of the VOIP traffic for procedures of file-sharing in a P2P network.

Keywords: traffic simulators, queuing networks, P2P networks, VOIP traffic

1. Introduction

A peer-to-peer network is a computer net that does not have a hierarchy of nodes fixed before. All the network nodes are equivalent, peers [5]. They are both clients and servers for the other nodes of the net. The P2P networks have been developed for music and multimedia file-sharing. Napster was one of the first file-sharing programmes. Gnutella is based on a widespread peer-to-peer network without central servers. The most famous decentralized client is Kazaa. eMule is the most used peer to peer software nowadays.

A new frontier of P2P network employment is related to VOIP applications. The most popular VOIP telephony application on the P2P network is Skype.

The VOIP technology makes it possible to make a telephone conversation using an Internet connection or another dedicated network that uses the IP
3.16

L. Pasini and S. Feliziani

Figure 1. Employment of VOIP technology

Figure 2. RTP package header

2. P2P Networks and VOIP applications

The architecture of a VOIP application requires contemporary presence of different typologies of protocols of communication to manage its different layers [6]. For instances the RTP protocol (Real-Time Transport Protocol) may be used to manage the data transport level, that is, the flows of voice packages on an IP contemporary to the SIP protocol (Session Initiation Protocol) which manages different sessions over which the VOIP communication is based.

The RTP protocol allows for distribution of services that need the data transfer in real time. This happens for example in case of audio and video interactivity among users connected by a P2P network. The four main fields of the header of an RTP data package are:

- load type,
- sequence number,
- time marking,
- source identification.

They are illustrated in Figure 2 and are described afterwards.

<table>
<thead>
<tr>
<th>Load Type</th>
<th>Sequence Number</th>
<th>Time Marking Number</th>
<th>Synchronization source identifier</th>
<th>Any other</th>
</tr>
</thead>
</table>

Figure 2. RTP package header
1. **Load type** (7 bit): identifies the employed encode.

2. **Sequence number** (16 bit): the sequence number is increased by one unit for every RTP package sent, and can be used by the receiver to notice the losses and reset the package sequence.

3. **Time marking** (32 bit): records the instant of the first byte sampling in the data package. The time marking comes from the transmitter’s sampling clock and the receiver can use it to remove the package jitter and supply a synchronized reproduction.

4. **Synchronization source identifier** (32 bit): identifies the RTP flow source. Usually every flow of an RTP session has its own SSRC (Synchronization Source Identifier). This identifier is not the IP address of the transmitter, but a number that the source assigns at random when a new flow initialises.

**The SIP protocol** is an application level protocol through which it is possible to establish, modify and end multimedia sessions. It is a standard protocol of the IETF (RFC 3261), independent from the used transport protocol. It is programmable and therefore easily extensible.

The SIP protocol is of the request-response type and it is modelled on other IETF protocols such as HTTP and SMTP. It has a reliable mechanism based on timeout. It can base itself both on transport protocols which may be connection oriented like TCP or not like UDP. SIP uses the RTP protocol for the transport of real-time data, the SDP protocol (Session Description Protocol) for the multimedia sessions description and the SAP protocol (Session Announcement Protocol) for the spreading of multicast communications. The scheme that illustrates the structure of the communication protocols pile is illustrated in Figure 3.

<table>
<thead>
<tr>
<th>Session Management</th>
<th>Media Agents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Session Setup and Discovery</td>
<td>Audio &amp; Video</td>
</tr>
<tr>
<td>SDP</td>
<td>RTP/RTCP</td>
</tr>
<tr>
<td>SIP</td>
<td>SAP</td>
</tr>
<tr>
<td>TCP</td>
<td>UDP</td>
</tr>
<tr>
<td>IP/ICMP and IP Multicast/ICMP</td>
<td></td>
</tr>
</tbody>
</table>

**Figure 3. Protocol pile**

The methods implemented by the SIP protocol are the following:

- **INVITE**: starts a call inviting a user to join in a session;
- **ACK**: confirms that the user received a final answer to a request;
- **BYE**: indicates the end of a call;
- **CANCEL**: deletes a pending request;
- **OPTIONS**: requires the available capacities of the server;
• REGISTER: records the user.

The answers are characterized by a Status-Code that is a number \( nxx \). The first character \( n \) defines the class of the answer. The possible classes of answer are explained below:

- **1xx**: Provisional indicates that a request has been received, but the contacted server does not have a final answer yet and is continuing to process the request;
- **2xx**: Success indicates that the request has been successful;
- **3xx**: Redirection indicates that there are new locations for the user and so other actions are necessary to complete the request;
- **4xx**: Request Failure indicates that the request contains a wrong syntax or cannot be completed by this server, it must be then modified and afterwards one can try again;
- **5xx**: Server Failure indicates that the server cannot complete an apparently valid request;
- **6xx**: Global Failure indicates that the request cannot be completed by any server.

In the request and answer messages there are the following fields:

- **Call-ID**: is the global identifier of calls or of the recordings of a particular client;
- **To**: contains the logic address of the person called, it may not be a definitive one because the called person can move in the network;
- **From**: identifies the user that started the request;
- **Contact**: contains a list of user addresses for the system;
- **Cseq**: identifies the voice call. It also contains the request method;
- **Via**: indicates the path made by the request;
- **Content-type**: indicates the used protocol;
- **Expires**: indicates the time validity of the recording of a user at the SIP Registrar Server.

The scheme of a triple handshake necessary for the establishment of the VOIP communication according to the SIP protocol is illustrated in succession.

In a project previously developed [3] we studied the performances of the P2P network illustrated in Figure 5 where the traffic was exclusively due to file-sharing procedures. The aim of this paper is to value how the executions of VOIP calls impact on the P2P network file-sharing data traffic performances.

In particular three different systems with the same network typology have been evaluated. The three systems distinguish themselves for the technology with which local networks are implemented to single super-nodes. The technologies taken into consideration are the following:

- LAN Fast Ethernet, with 100Mbps throughput;
- Wireless IEEE 802.11g, with 54Mbps throughput;
- ADSL2+ ITU G.992.5 M, with 3.5Mbps throughput.
3. Implementation of P2P network simulator

The basic object types defined for setting up P2P network simulators with VOIP traffic are listed in succession:

- **Peer**: represents a standard peer;
- **Rendez-Vous**: represents a peer group leader;
- **Link**: represents a network communication channel;
3.1. The object type peer

A peer is identifiable in a P2P network by a couple of integer numbers \((\text{UUID}, \text{GROUPID})\). UUID is the peer identifier inside a group of belonging that is identified by GROUPID.

In Figure 6 we illustrate the logic functional structure of a peer in this new context.

**Figure 6. Logic-functional structure of a Peer**

Starting from the logic-functional structure, we have defined the architecture of a Peer component in the P2P network. This architecture is based on the use of queue service systems. It consists in a network of single server queues. The P2P network messages are represented as customers in the queuing network that are served basing on their class.

The architecture of a Peer object is illustrated in Figure 7.

The object type Peer contains some queue type variables on which the following single server stations are structured:

- **USER**: is the source of the local traffic. It generates, through a \texttt{BOOT} procedure, the \texttt{ADVREQ} type messages. These messages represent the requests of the Peer to record on the P2P network and are sent to the \texttt{CONTROL} queue.
- **CONTROL**: is a station with a finite capacity. Here, the \texttt{ADVREQ} messages are forwarded to the \texttt{PROTOCOL} queue that is the core of the P2P application.
- **PROTOCOL**: this station serves customers with a priority scheduling which is pre-emptive. It serves the entering of \texttt{DATA\_MSG} and \texttt{VOICE\_MSG} type messages (respectively with the \texttt{DOWNLOAD} and \texttt{VOICEOUT} classes) and the outgoing ones...
Figure 7. Architecture of a Peer in the P2P network

(respectively with the UPLOAD and VOICEIN classes). The service procedures are in the order: PR_IN, PR_VCIN (for the entering messages) and PR_OUT, PR_VCOUT (for the outgoing messages).

The PR_IN procedure processes the DATA_MSG messages that come from the network. If the received message has the DESTID and DGID fields corresponding to the UUID and the GROUPID of the receiving Peer, respectively, the following cases appear, basing on the value of the type field of the message:

1. ADVACK: this case corresponds to the confirmation of a recording that has already happened by one Rendez-Vous. The Peer activates the control procedure that verifies the existence of a voice call in the calls vector. In case of a positive result, it sets a timer that will launch the CR_INV procedure that simulates the starting phase of the voice call. In case of a negative result, the Peer executes the DATA procedure. This procedure generates messages of file requests, FILEREQ, towards other Peers in the network.
2. FILEREQ: this case corresponds to the request of a file by a remote Peer. The Peer sends a FILEREQ type message towards the asking Peer through the SENDFILE procedure.
3. FILERES: this case corresponds to the reception of a message containing a resource previously requested to a remote Peer. The Peer forwards it to its storage device represented by the HD queue.

The PR_OUT procedure processes the DATA_MSG type messages outgoing from the Peer towards the P2P network. These messages can be of the following types:
1. ADVREQ, that indicates a request of a recording at the Rendez-Vous;
2. FILEREQ, that indicates a file request towards a network Peer;
3. FILERES, that indicates the sending of a file towards a requesting Peer.

After the processing, the procedure sends the messages in the SOCKET queue of the Peer.

The PR_VCIN procedure processes the VOICE_MSG messages entering the Peer. The following events, based on the types of messages, can take place:
1. Reception of the invitation to establish an audio connection with a remote Peer. In this case the request is accepted if the Peer state is not busy and is sent an OK message.
2. OK Reception by a remote Peer. In this case the Peer STATCODE code is controlled, an ACK message is generated and the service procedure of the VOICE queue is activated.
3. ACK reception by a remote Peer. In this case there is a confirmation that the connection for a voice call has been established. The CAP_BAND parameter is updated by the net links used by the voice call and it starts to generate the answer audio flow activating the VOICE queue service.
4. Reception of BYE from a remote Peer. This case corresponds to the event of closing a voice call by the remote Peer. The generation of the audio flow is blocked, an OKCLOSE message is sent to confirm the end of the voice call and the CAP_BAND parameters of the links, used for the voice call, are updated.
5. Reception of the OKCLOSE by a remote Peer. This case corresponds to the last event of a voice call closing.
6. Reception of an AUDIO type message. This message is sent to the Peer audio card, that processes the audio messages for the output device.

The PR_VCOUT procedure processes the messages of a VOICE_MSG type related to the audio flow outgoing from the Peer. The following cases may take place according to the type of message entering the PROTOCOL queue:
1. Reception of an AUDIO type message. This message is sent to the SOCKET queue.
2. Reception of a BYE type message. The message is sent to the SOCKET queue to be forwarded in the network.

• CACHE: it is the medium for the temporary memorization of the copies of the messages sent to the PROTOCOL queue on the network waiting for the reception confirmation. If at the Time Out expiry, a reception confirmation of the message is not received, this last message is inserted again in the PROTOCOL queue to be transmitted again in the network.
Simulators of VOIP Traffic in P2P Networks

- **HD**: it serves the FILERES type messages received by remote Peers. It memorizes the messages coming from the PROTOCOL queue inside itself.

- **SOCKET**: it represents the device that interfaces the Peer with the network. It processes both the DATA_MSG and the VOICE_MSG type messages outgoing from the Peer, that have respectively UPLOAD and VOICEIN classes, and the DATA_MSG and VOICE_MSG entering the Peer that have respectively the DOWNLOAD and VOICEOUT classes. The outgoing messages are processed with a speed proportional to the outgoing band, UP_R and those entering with a speed proportional to the entering band, DOWN_R. The messages outgoing from the Peer are forwarded from the SOCKET queue to the CHANNEL queue of the network channel where the Peer is connected to. The DATA_MSG type messages are forwarded to the network with the UPLOAD class while those of the VOICE_MSG type with the VOICEIN class.

- **VOICE**: generates the audio flow outgoing from the Peer. In particular the CREATEVC procedure generates a flow of audio packages for a slot equal to the duration of the conversation. These packages are sent into the AUDIOCRD queue with a VOICEIN class. In the setting up of audio messages, the TRACERT table saved as an attribute in the message responsible of the call establishment is used. At the end of the conversation the CREATEVC procedure generates a BYE type message with the VOICEIN class.

- **AUDIOCRD**: represents the device of audio processing. The outgoing VOICE_MSG messages are sent to the PROTOCOL queue in the VOICEIN class. The entering VOICE_MSG messages are processed and sent to the OUTPUT queue in the VOICEOUT class.

- **OUTPUT**: queue represents the Peer output devices, receivers of the VOICE_MSG messages coming from the AUDIOCRD queue.

The definition code of the object type Peer is reported as follows. The programming code is developed in this paper using the QNAP2.V9 language [7].

```c
OBJECT PEER(UUID,GROUPID);
    INTEGER UUID;
    INTEGER GROUPID;
    INTEGER IDRV;
    REAL T_ARR;
    INTEGER TIME_OUT;
    REAL CPURATE, HDRATE,
    REAL UP_R,DOWN_R;
    REAL FSTORE;
    QUEUE USER, PROTOCOL,CONTROL;
    QUEUE SOCKET;
    QUEUE AUDIOCRD, OUTPUT, VOICE;
    REF QUEUE NETWORK;
    QUEUE AUDIOCRD, OUTPUT, VOICE;
    INTEGER VOICETR(MAX_HOPS,2);
    INTEGER ROLE
    REAL OUTST;
    REAL OUTRATE;
    BOOLEAN CONTROLB;
    BOOLEAN ISACTIVE;
```
Referring to the previous study [3] we shortly describe the variables added for the VOIP traffic:

- **CONTROLB**: indicates if a Peer is busy in a voice conversation;
- **ISACTIVE**: shows the availability of the Peer to accept voice calls;
- **INITPTR**: is used to implement the audio messages routing;
- **SETTIME**: is used to run the timer that regulates the start of a voice call;
- **CALL**: is a pointer used to access the array of the P2CALL;
- **VOCE**: is a timer that activates the voice call;
- **OUTST**: indicates the number of bytes processed by the audio output device during the simulation;
- **OUTRATE**: shows the capacity of the output device, expressed in bits/ms;
- **ROLE**: this variable can assume the value 1 or 2. The value 1 means that the Peer is the receiver of a voice call in the model. The value 2 shows that the Peer is the sender of a voice call;
- **VOICETR**: is an array that contains the information to forward the audio messages to the Peer receiver of a call in the unicast mode. This array is initialised in the establishment phase of the voice call between two Peers.

The definition code of the template stations in the new configuration is reported as follows:

```plaintext
/STATION/
NAME = *PEER.PROTOCOL;
SCHED = PRIOR,PREEMPT;
SERVICE(UPLOAD) = PR_OUT(CPURATE);
SERVICE(DOWNLOAD) = PR_IN(CPURATE,UUID,GROUPID);
SERVICE(VOICEIN) = PR_VCIN(CPURATE);
SERVICE(VOICEOUT) = PR_VCON(CPURATE,UUID,GROUPID);
TRANSIT = OUT;

/STATION/
NAME = *PEER.SOCKEET;
SCHED = PRIOR,PREEMPT;
SERVICE(UPLOAD) = BEGIN
  EXP(DATA_MSG.SIZE/UPLOAD_R);
  TRANSIT(NETWORK,UPLOAD);
END;
SERVICE(DOWNLOAD) = BEGIN
  EXP(DATA_MSG.SIZE/DOWNLOAD_R);
  TRANSIT(PROTOCOL,DOWNLOAD);
END;
SERVICE(VOICEIN) = BEGIN
  EXP(VOICE_MSG.SIZE/UPLOAD_R);
  TRANSIT(INCLUDIN(QUEUE)::PEER.NETWORK,VOICEIN,2);
END;
SERVICE(VOICEOUT) = BEGIN
  EXP(VOICE_MSG.SIZE/DOWNLOAD_R);
```
TRANSIT(INCLUDIN(QUEUE)::PEER.PROTOCOL, VOICEOUT, 2);
END;
TRANSIT = OUT;

/STATION/
NAME = *PEER. VOICE;
SERVICE = BEGIN
CREATEVC (UUID, GROUPID, VOCE_MSG.SENDID, VOCE_MSG.SGID);
END;
TRANSIT = OUT;

/STATION/
NAME = *PEER.AUDIOCRD;
SERVICE(VOICEIN) = BEGIN
EXP(VOCE_MSG.SIZE/CPURATE);
TRANSIT(INCLUDIN(QUEUE)::PEER.PROTOCOL, VOICEIN, 2);
END;
SERVICE(VOICEOUT) = BEGIN
EXP(VOCE_MSG.SIZE/CPURATE);
TRANSIT(INCLUDIN(QUEUE)::PEER.OUTPUT, 2);
END;
TRANSIT = OUT;

/STATION/
NAME = *PEER.OUTPUT;
SCHED = FIFO;
SERVICE = BEGIN
EXP(VOCE_MSG.SIZE/OUTRATE);
OUTST := OUTST + VOCE_MSG.SIZE;
END;
TRANSIT = OUT;

3.2. The Rendez-Vous object type

The Rendez-Vous object of the P2P network has the following facilities:

- it records Peers in the network,
- it executes the routing of messages in the network,
- it verifies the availability of the communication channels in the INVITE phase of a voice call.

The logic-functional structure of the Rendez-Vous is illustrated in Figure 8.

Let us assume that every Peer is directly connected to a Rendez-Vous and this connection remains active for the duration of the simulation. We have fixed the maximum time of simulation for every experiment at 500 seconds. So that we assume that the network configuration remains the same during this slot. Each Rendez-Vous is connected to other Rendez-Vous objects implementing in this way a network interconnection among more heterogeneous networks. The software layer implemented in the Rendez-Vous keeps trace of different simultaneous connections represented by the internal socket objects. Every Rendez-Vous has a UUID univocal identifier in the P2P network. The software layer of this object contains a Peer mode management that memorizes the network recording requests received by single Peers. The data structure used is a table named PTable formed
Figure 8. Logic-functional structure of a Rendez-Vous

by triples of integers of a [PeerID, PeerGID, LinkID] type that identify the connection to the single Peers.

The following service queue devices with a single server have been singled out in the Rendez-Vous object architecture (Figure 9).

**PROTOCOL:** this service station has a scheduling based on priority levels with pre-emption. It serves the DATA_{MSG} and VOICE_{MSG} messages entering the Rendez-Vous, respectively with \textsc{DOWNLOAD} and \textsc{VOICEOUT} classes. The outgoing DATA_{MSG} messages are managed using the UPLOAD class. The service procedures for these messages are \textsc{RVP\_DOWN}, \textsc{RVP\_OUT} and \textsc{RVP\_UP}, respectively.

The \textsc{RVP\_DOWN} procedure executes a processing time that depends on the message dimension and it controls if the message corresponds to the recording request sent by a Peer to the Rendez-Vous or not. Therefore, the following cases appear:

- The message corresponds to the recording request at the Rendez-Vous. In this case, the TYPE field of the message is equal to \textsc{ADVREQ}. After the recording of the requesting Peer with the \textsc{REG\_PEER} procedure, through the \textsc{SADVACK} procedure, an \textsc{ADVACK} type message is sent in the PROTOCOL queue with UPLOAD class. The \textsc{ADVACK} message is sent exactly to report the recording that has already happened to the sender Peer.

- The message corresponds to a request of a file to be forwarded in the unicast mode in the network. In this case, the message is inserted again in the PROTOCOL queue in the UPLOAD class with a high level priority.
The **RVPV_OUT** procedure manages the messages related to the audio flow entering the PROTOCOL of the Rendez-Vous. First of all, it is controlled that the receiver-Peer does not correspond to the sender-Peer. In this case the message is rejected. Otherwise, the COUNT variable is increased. Should COUNT reach the value of the constant MAX_HOPS, the message is refused because it is avoided that it enters a loop in the network. If the Rendez-Vous receives an OK message with STATCODE equal to 200, it controls that there is the minimum band necessary to convey a voice call to the link. All the other messages entering the PROTOCOL of the Rendez-Vous that are not rejected because of the above mentioned reasons, are sent to the Rendez-Vous SOCKET with the VOICEIN class.

The **RVP_UP** procedure executes a processing time that depends on the message dimension and forwards it to the SOCKET queue with the UPLOAD class.

**SOCKET**: this service station has a scheduling based on a priority level which is pre-emptive. The SOCKET queue manages the entering and outgoing messages of the network, interfacing with the Link layer. The entering messages of the DATA_MSG and VOICE_MSG type have the DOWNLOAD and VOICEOUT classes, respectively and the messages outgoing of the DATA_MSG and VOICE_MSG type have the UPLOAD and VOICEIN classes, respectively. The service procedures are described below.

The **RVSIN** procedure executes the message reception with the speed proportional to the reception band and forwards it to the PROTOCOL queue with the DOWNLOAD class.
The RVSVIN procedure executes the audio message reception with the speed proportional to the reception band and forwards it to the PROTOCOL queue with the VOICEOUT class.

The RVSVOUT procedure executes the physical transmission of the message and controls its typology. If the message is a general request of the FILEREQ or ADVREQ type, the UUID of the current Rendez-Vous is inserted in the last position of the TRACERT table of the message. In this way, at the request phase, the necessary information to the route that the answer will follow in the network is memorized. In particular, if the message is destined to a peer directly connected and recorded in the PTable of the current Rendez-Vous, the message is routed to the link corresponding to that Peer. On the contrary, the network transmission procedure is executed:

1. The type of message is FILEREQ. The Rendez-Vous forwards copies of the message received, only to the other connected Rendez-Vous objects, except for that from which the message is coming. The link connected to the current Rendez-Vous on which it is possible to forward the messages is relieved through the CANSEND function that finds out all the links towards the Rendez-Vous objects of the net.

2. The type of message is FILEREQ. The link on which the message is forwarded is determined in the FORWARD procedure, scrolling backwards the entry table TRACERT of the message, compiled by the Rendez-Vous crossed in the transmission phase of the request.

The RVSVOUT procedure manages the messages of the VOICE_MSG type outgoing from the SOCKET queue. The task of the SOCKET queue is to address the message towards the outgoing link. First of all, it is verified if the message is of the INVITE type. In this case, as the message is sent in the broadcast mode, the crossed Rendez-Vous ID is saved. Then, it is verified with PCONN procedure if the receiver’s Peer is directly connected to the Rendez-Vous. If the Peer is not directly connected to the Rendez-Vous, the following situations may take place basing on the value of the TYPE field of the message:

1. INVITE: the message is sent in the broadcast modality to all the possible links. The links are determined on the CANSEND procedure. The messages are sent to all the links through they have not gone yet.

2. OK or OKCLOSE: the POINTER value is reduced and the message is sent to the identified link through the FORWARD procedure. This procedure uses the entry point of the TRACERT table.

3. ACK or BYE: the POINTER value is increased and the message is sent to the link identified through the FORWARD procedure.

4. AUDIO: the Peer’s role is established if the POINTER must be increased (sender) or reduced (receiver).

The LinkFD devices that are illustrated in Figure 9 represent the Rendez-Vous connections with the transmission channels of the P2P network. A procedure
executes the connection between a Rendez-Vous and the LinkFD associated to it, in the moment of the setting up of the P2P network. This procedure properly executes the assignment of values to some queue pointer variables that are present as internal variables of the object.

A P2P network Rendez-Vous is implemented with an object named RV that is set up with two formal parameters UUID and N_LINKS. These parameters represent the Rendez-Vous identifier in the P2P network and the number of links to which it is connected, respectively. The definition code of the object is reported here as follows:

```plaintext
OBJECT RV(UUID,N_LINKS);
  INTEGER UUID;
  INTEGER N_LINKS;
  REAL CPURATE;
  REAL UPR,DOWNR;
  QUEUE PROTOCOL;
  QUEUE SOCKET;
  INTEGER ID_LINKS(N_LINKS);
  REF QUEUE LINK_IO(N_LINKS);
  INTEGER PTABLE(MAXPEERS,3);
  INTEGER PPOINTER;
  INTEGER C_VO_INV;
  INTEGER C_VO_OK;
  INTEGER C_VO_ACK;
  INTEGER C_VO_BYE;
  INTEGER C_VO_OKC;
  INTEGER C_VO_AUD;
END;
```

Among the variables present in the definition of the Rendez-Vous object type, we find the following variables for the management of the traffic flows:

- **CPURATE**: indicates the RV computing capacity;
- **UPR** and **DOWNR**: indicate the networking capacities and are measured in bit/ms.

Two tables are also present in the definition:

- **PTABLE**: Peer recording table;
- **ID_LINKS**: this table is implemented using the array of integer ID_LINKS and the queue pointers array LINK_IO. The two arrays have the same length and the ID_LINKS(n) supplies the identifier, in the P2P network, of the LinkFD object to which the queue pointer LINK_IO(n) points.

The definition code of the internal queues of the Rendez-Vous object is reported here as follows:

```plaintext
/STATION/
  NAME = RV_PROTOCOL;
  SCHED = PRIOR,PREEMPT;
  SERVICE(UPLOAD) = RVP_UP(CPURATE);
  SERVICE(DOWNLOAD) = RVP_DOWN(UUID,CPURATE,PTABLE,PPOINTER);
  SERVICE(VOICEOUT) = RVPV_OUT(CPURATE);
  SERVICE(VOICEIN) = CST(0);
  TRANSIT = OUT;
```
L. Pasini and S. Feliziani

```plaintext
/STATION/
NAME = *RV_SOCKET;
SCHED = PRIOR, PREEMPT;
SERVICE(UPLOAD)=RVSOUT(UUID,DATA.MSG.SIZE,UPR,N_LINKS,ID_LINKS, PTABLE,PPOINTER);
SERVICE(DOWNLOAD) = RVSIN(DATA.MSG.SIZE,DOWNR);
SERVICE(VOICEIN)=RVSVOUT(UUID,VOICE.MSG.SIZE,UPR,N_LINKS,ID_LINKS, PTABLE,PPOINTER);
SERVICE(VOICEOUT)=BEGIN
  EXP(VOICE.MSG.SIZE/DOWNR);
  TRANSIT(INCLUDIN(QUEUE)::RV.PROTOCOL,VOICEOUT,2);
END;
TRANSIT = OUT;
```

3.3. **LINKFD object type**

This object type represents the full-duplex transmission channels of the P2P network (Figure 10).

Basing on the used technology, it can be either of the wired or wireless type.

![Figure 10. Logic functional structure of a Link](image)

The **LINKFD** object of the P2P network contains two variables of queue type: **CHANNEL1** and **CHANNEL2** that have the following structure.

**CHANNEL:** it is an infinite servers queue station. It simulates a one-way link in a communication channel of the P2P network. It executes the change of class of the messages that cross it as it is shown in Figure 10. The service procedures for the type of messages DATA_MSG and VOICE_MSG are STM_DATA and STM_VOICE, respectively.

The definition code of this object type is reported here as follows:

```plaintext
OBJECT LINKFD(ID);
  INTEGER ID;
  INTEGER LOSTMSG;
  REAL LOSTSIZE;
  REAL T,P_ERR;
  QUEUE CHANNEL1;
  QUEUE CHANNEL2;
```
Every LINKFD channel of the P2P network has an integer identifier \(\text{ID}\) in the network. The channel band width is \(\mathcal{T}\), and it is measured in bit/ms. The integer variable \(\text{LOSTMSG}\) memorizes the total number of the messages lost during the transmission on the channel and the \(\text{LOSTSIZE}\) variable supplies the amount of the bit corresponding to such losses. The probability of losing the message is expressed by the internal real variable \(\text{P\_ERR}\).

The definition code of the LINKFD template stations is reported here as follows:

```plaintext
/STATION/
NAME = *LINKFD.CHANNEL1;
TYPE = INFINITE;
SERVICE(VOICEIN) = BEGIN
   STM_VOCE(P\_ERR, T, ID, LOSTVOCE, LOSTSVOC);
   TRANSIT(END1, VOICEOUT);
END;
SERVICE(UPLOAD) = BEGIN
   STM_DATA(P\_ERR, CAP\_BAND, ID, LOSTMSG, LOSTSIZE);
   TRANSIT(END1, DOWNLOAD);
END;
/STATION/
NAME = *LINKFD.CHANNEL2;
TYPE = INFINITE;
SERVICE(VOICEIN) = BEGIN
   STM_VOCE(P\_ERR, T, ID, LOSTVOCE, LOSTSVOC);
   TRANSIT(END2, VOICEOUT);
END;
SERVICE(UPLOAD) = BEGIN
   STM_DATA(P\_ERR, CAP\_BAND, ID, LOSTMSG, LOSTSIZE);
   TRANSIT(END2, DOWNLOAD);
END;
```

### 3.4. Object type messages

We have defined two subtypes of the CUSTOMER object type: the \textit{DATA\_MSG} and the \textit{VOICE\_MSG}. These messages respectively constitute the data traffic and the voice traffic in the P2P network.

- **DATA\_MSG**: its structure is represented in Figure 11. In the structure the following fields can be recognized:
  - \textit{SENDID}: integer identifier of the sending peer;
  - \textit{SGID}: integer identifier of the group to which the sending peer belongs;
  - \textit{DESTID}: integer identifier of the group to which the sender’s peer belongs;
  - \textit{TYP}: description string of the message. It can assume the values \textit{AdvRequest}, \textit{AdvAck}, \textit{FileReq}, \textit{FileRes};
  - \textit{SIZE}: dimension of the message expressed in a bit;
Figure 11. Structure of the DATA.MSG object type

- **COUNT**: counter of the hop number carried out by the message in its route towards the destination;
- **POINTER**: pointer to a record of the TRACERT table;
- **TRACERT**: table containing the information regarding the path followed by the message towards the destination. Every record of the table is composed by the couple (IDLink, IDRV) where the IDLink is the identifier of a Link crossed while IDRV is the identifier of the Rendez-Vous on which the Link IDLink goes out in the transit direction.

The definition code of this object type is reported as follows:

```c
CUSTOMER OBJECT DATA_MSG;
    INTEGER SENDID, DESTID, SGID, DGID;
    STRING TYP;
    REAL SIZE;
    INTEGER COUNT;
    INTEGER POINTER;
    INTEGER TRACERT(MAX_HOPS,2);
END;
```

Figure 12. Structure of the VOICE_MSG object type
VOICE_MSG: the structure of this object type is represented in Figure 12. In the structure the following additional fields regarding DATA_MSG can be recognized:

- **PAYTYP**: 7 bit – identifies the employed code;
- **SEQNUM**: 16 bit – identifies the position of the message inside a message sequence. It is used by the receiver to find out the losses and recreate the package sequences;
- **TIMEST**: 32 bit – reports the instant of the sampling of the first byte in the data package;
- **SRCID**: 32 bit – identifies the source of the audio flow. It is a number that is assigned by the source when a new audio flow is initialised;
- **STATCODE**: integer code of 3 characters that identifies the answer code of a peer. We take into consideration the following values for this field [*]:
  - 200: *Success* shows that the request has been successful,
  - 501: *Failure* indicates that the peer that has received the request is busy,
  - 502: *Failure* shows that the peer that has received the request is not active,
  - 503: *Failure* indicates that the Rendez-Vous has verified the lack of the minimum band.

In this case the TYP field of the message can have *Invite, Ok, OkClose, Ack, Audio* and *Bye* as a value.

The definition code of this object type is reported here as follows:

```
CUSTOMER OBJECT VOICE_MSG;
    INTEGER SENDID, DESTID, SGID, DGID;
    INTEGER RUOLO;
    STRING TYP;
    REAL SIZE;
    INTEGER COUNT;
    INTEGER POINTER;
    INTEGER TRACERT(MAX_HOPS, 2);
    INTEGER PAYTYP;
    INTEGER SEQNUM;
    INTEGER TIMEST;
    INTEGER SRCID;
    INTEGER STATCODE;
END;
```

4. P2P network and VOIP protocol

In this paragraph we describe the protocol procedures that form the VOIP communication sessions among the components of the P2P network. The reference protocol is SIP. In the SIP protocol the phases of a VOIP communication on a P2P network are the following:

- start of the VOIP communication;
- VOIP communication;
- closing of the VOIP communication.

Let us describe each of these phases in detail.
4.1. Start of a VOIP communication

The procedure to start a VOIP communication between two Peers on the P2P network is divided into the three following procedures:

- INVITE,
- OK,
- ACK.

We describe each of these procedures using exemplifying schemes.

**INVITE** procedure: in this phase the Peer of the P2P network that wants to start a VOIP communication sends an **INVITE** message towards the remote Peer with which it wants to communicate.

Figure 13 illustrates a diagram of the modes with which the Peer (1,1) sends an **INVITE** message to the Peer (2,2) of the P2P network.

The **INVITE** procedure is divided into 5 phases that are described here as follows using the scheme of Figure 13 as an example:

1. The Peer (1,1) invites the Peer (2,2) to start a voice call forwarding the invite message to its own Rendez-Vous. The receiver of this message is Peer (2,2).
2. The Rendez-Vous spreads the request in the network in a broadcast mode through the “Reverse Path Forwarding” algorithm, memorizing in the TRACERT table of the message, the necessary information for the answer that will be transmitted backward in a unicast mode.
3. The Rendez-Vous 2 in which the receiving Peer (2,2) is recorded forwards the INVITE message to this last one.

4. The Peer (2,2) that has received the INVITE message can answer it in three different modes, as illustrated subsequently. In all three cases the receiving Peer answers by sending an OK message to the sending Peer.

   The OK message contains the classification code of the answer:

   • do not accept the conversation request (code 501),
   • busy, not accepting the voice call (code 502),
   • accept the request of conversation (code 200).

   OK procedure: with this procedure the Peer called forwards the OK message to its own Rendez-Vous, enclosing the answer code in the message header.

   Figure 14 and the comment that follows schematically describe this phase of the procedure to establish a VOIP call.

---

Figure 14. OK Connection Procedure

1. Peer (2,2) sends an OK message to Rendez-Vous 2 specifying Peer (1,1) as the addressee. In case the Peer does not accept to start a conversation, the Rendez-Vous sends an OK message with code 501 or 502 again without controlling the channel availability;

2. All the Rendez-Vous that are in the path of the answer that is transmitted in unicast mode, verify, in case the call request is accepted, the channel availability to reserve the transmission band capacity necessary to guarantee good quality of communication;
3. In case this availability is not guaranteed, the Rendez-Vous will communicate to both the sender and the receiver the impossibility of establishing a connection.

When the calling Peer receives an OK answer message with code 200 from the Peer called, it sends an ACK message to confirm the communication.

ACK procedure: Figure 15 and the comment that follows describe this procedure to establish a VOIP communication.

![Figure 15. ACK Connection Procedure](image)

1. Having received an OK (with code 200), Peer (1,1) forwards an ACK to its own Rendez-Vous;
2. The Rendez-Vous spreads the ACK on the network in unicast mode using the message Tracert;
3. Peer (2,2), that is the receiver, memorizes the ACK and it is ready for a voice call.

4.2. **VOIP communication**

The connection having been established, the two Peers start a VOIP communication. The full-duplex transmission channel structure, implemented by the two transmission channels of the LinkFD objects of the P2P network is used in this phase.
During the communication phase, the quality of the service requires that the minimum necessary requirements are maintained to develop the VOIP communication.

Figure 16 and the comment that follows describe this VOIP communication phase in a P2P network.

**Figure 16.** Audio message transmission phase

1. The connection having been established, Peer (1,1) keeps two link channels for full-duplex communication of audio messages. A channel for reception and another channel to send audio packages are reserved;
2. The connection having been established, Peer (2,2) keeps also the two link channels for full-duplex communication of audio messages;
3. The duration of the communication was pre-defined in the initialization phase of the VOIP call.

### 4.3. Closing of VOIP communication

The closing of a VOIP communication is divided into two procedures:

- **BYE**,
- **OK**.

**BYE** procedure: the calling Peer sends the receiving Peer a **BYE** message to close the voice communication. Figure 17 and the following comment illustrate how this procedure develops.
Figure 17. BYE Closing Phase

Figure 18. OK Closing Phase
1. Peer (1,1) sends Peer (2,2) a BYE message to end the VOIP communication;
2. The Rendez-Vous spreads the BYE message in unicast mode using the message TRACERT table;
3. Peer (2,2) receives the BYE message and memorizes it inside its own storage device for processing.

OK procedure: the BYE message having been received by the calling Peer, the Peer called answers the last peer sending him an OK message. Figure 18 and the following comment illustrate how the procedure develops.
1. Having received the BYE message, Peer (2,2) closes the call. It updates the network capacity availabilities of the channels used by the VOIP communication, increasing them in terms of quantity in respect of the transmission band previously occupied and sends Peer (1,1) an OK message to confirm the closing.
2. Rendez-Vous 2 spreads the OK message in unicast mode, using the message TRACERT table.
3. Peer (1,1) receives the OK message and stops the VOIP message flow.

5. Simulation of traffic in P2P network

In this paragraph the results of the study of performance in the P2P network that is shown in Figure 19 are described. In this study the network typology was left unchanged and simulators in the P2P network with three different configurations of the local networks afferent to each Rendez-Vous were constructed.

The configurations of the local networks were characterized as follows:

![Diagram of P2P network with VOIP communications](image_url)
LAN Fast Ethernet, with 100 Mbps throughput,
Wireless IEEE 802.11g, with 54 Mbps throughput,
ADSL2+ ITU G.992.5 M, with 3.5 Mbps throughput.

As far as the VOIP traffic in the network is concerned, activations of the three VOIP communications that are described in the following scheme were assumed. The duration of each simulated call was fixed at 500,000 ms.

- Peer (4,1) calls Peer (2,2). The call starts at 15,000 ms of the simulation and has a duration of 300,000 ms,
- Peer (2,1) calls Peer (3,1). The call starts at 50,000 ms of the simulation and has a duration of 250,000 ms,
- Peer (1,1) calls Peer (5,1). The call starts at 35,000 ms of the simulation and has a duration of 100,000 ms.

**Performance parameters**: Our target was to value how the network performance was affected by the presence of the voice traffic characterized by a higher priority compared to the data traffic. Different parameters were measured basing on the types of objects present in the P2P network. The parameters considered with regard to the Peer and Rendez-Vous object types were the following:

- Protocol MResponse: average response time in the queue PROTOCOL,
- Protocol MService: average service time in the queue PROTOCOL,
- Socket MResponse: average response time in the queue SOCKET,
- Socket MService: average service time in the queue SOCKET,
- Protocol Nbout: number of customers served by the queue PROTOCOL,
- Socket Nbout: number of customers served by the queue SOCKET.

**5.1. Results of simulations**

The results of simulations related to the three examined types of P2P networks are reported in succession. For each net type, the results are subdivided into two tables:

- Peer tables,
- Rendez-Vous tables.

The performance parameters computed from the last test conducted on the system during the simulation are displayed in the tables. These measurements are carried out relatively to the last 50 sec period of the simulation run.

**5.1.1. P2P network with file-sharing + VOIP traffic**

The following tables illustrate the performance indexes related to the components of a P2P network where both the data traffic provoked by the file-sharing procedures active on the network and the traffic of audio communication related to the VOIP communications previously described are present. In this system, the VOIP traffic satisfies the Quality of Service conditions [5] and the three voice calls occur regularly. The priority level of the VOICE_MSG messages is higher than that of the DATA.MSG messages related to the file-sharing procedures.
Table 1. Fast Ethernet technology, Peers

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>105.0</td>
<td>12.78</td>
<td>2.241</td>
<td>1.412</td>
<td>12.78</td>
<td>2.396</td>
<td>1.361</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>105.7</td>
<td>14.74</td>
<td>1.479</td>
<td>1.416</td>
<td>14.74</td>
<td>2.651</td>
<td>1.386</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>112.2</td>
<td>14.48</td>
<td>8.664</td>
<td>8.166</td>
<td>14.48</td>
<td>2.432</td>
<td>1.318</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>137.6</td>
<td>15.74</td>
<td>4.528</td>
<td>4.250</td>
<td>15.74</td>
<td>4.674</td>
<td>1.371</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>106.3</td>
<td>15.29</td>
<td>8.808</td>
<td>8.323</td>
<td>15.29</td>
<td>2.474</td>
<td>1.365</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>168.3</td>
<td>15.31</td>
<td>2.451</td>
<td>2.337</td>
<td>15.31</td>
<td>2.592</td>
<td>1.532</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>115.5</td>
<td>14.98</td>
<td>2.248</td>
<td>2.141</td>
<td>14.98</td>
<td>2.531</td>
<td>1.368</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>123.5</td>
<td>13.96</td>
<td>9.260</td>
<td>8.697</td>
<td>13.96</td>
<td>2.647</td>
<td>1.457</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>109.9</td>
<td>12.77</td>
<td>1.553</td>
<td>1.487</td>
<td>12.77</td>
<td>2.373</td>
<td>1.410</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>141.3</td>
<td>15.32</td>
<td>4.692</td>
<td>4.436</td>
<td>15.32</td>
<td>3.583</td>
<td>1.421</td>
</tr>
</tbody>
</table>

Table 2. Fast Ethernet technology, Rendez-vous

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>5</td>
<td>116.4</td>
<td>2.140</td>
<td>2.131</td>
<td>116.4</td>
<td>24.17</td>
<td>1.388</td>
</tr>
<tr>
<td>101</td>
<td>3</td>
<td>73.5</td>
<td>1.174</td>
<td>1.170</td>
<td>73.5</td>
<td>7.78</td>
<td>1.124</td>
</tr>
<tr>
<td>102</td>
<td>5</td>
<td>117.5</td>
<td>1.939</td>
<td>1.934</td>
<td>117.5</td>
<td>12.41</td>
<td>1.228</td>
</tr>
<tr>
<td>103</td>
<td>5</td>
<td>112.0</td>
<td>1.368</td>
<td>1.364</td>
<td>112.0</td>
<td>37.60</td>
<td>1.327</td>
</tr>
<tr>
<td>104</td>
<td>2</td>
<td>57.1</td>
<td>1.258</td>
<td>1.254</td>
<td>57.1</td>
<td>3.29</td>
<td>0.801</td>
</tr>
</tbody>
</table>

Table 3. Wireless technology, Peers

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>115.8</td>
<td>13.30</td>
<td>2.281</td>
<td>2.188</td>
<td>13.29</td>
<td>4.659</td>
<td>2.581</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>109.9</td>
<td>15.18</td>
<td>1.491</td>
<td>1.429</td>
<td>15.18</td>
<td>4.936</td>
<td>2.590</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>109.6</td>
<td>14.16</td>
<td>8.621</td>
<td>8.193</td>
<td>14.16</td>
<td>4.382</td>
<td>2.458</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>121.1</td>
<td>14.69</td>
<td>4.500</td>
<td>4.275</td>
<td>14.69</td>
<td>5.377</td>
<td>2.605</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>106.3</td>
<td>15.51</td>
<td>8.716</td>
<td>8.318</td>
<td>15.51</td>
<td>4.791</td>
<td>2.536</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>160.5</td>
<td>15.01</td>
<td>2.440</td>
<td>2.330</td>
<td>15.01</td>
<td>4.706</td>
<td>2.802</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>119.2</td>
<td>15.28</td>
<td>2.222</td>
<td>2.119</td>
<td>15.28</td>
<td>4.783</td>
<td>2.568</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>121.6</td>
<td>13.74</td>
<td>8.924</td>
<td>8.505</td>
<td>13.74</td>
<td>4.546</td>
<td>2.589</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>118.4</td>
<td>13.26</td>
<td>1.558</td>
<td>1.491</td>
<td>13.26</td>
<td>4.566</td>
<td>2.691</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>157.4</td>
<td>15.88</td>
<td>4.743</td>
<td>4.520</td>
<td>15.88</td>
<td>5.611</td>
<td>2.671</td>
</tr>
</tbody>
</table>
Table 4. Wireless technology, Rendez-Vous

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>5</td>
<td>117.5</td>
<td>2.145</td>
<td>2.138</td>
<td>117.5</td>
<td>80.34</td>
<td>2.573</td>
</tr>
<tr>
<td>101</td>
<td>3</td>
<td>75.2</td>
<td>1.164</td>
<td>1.163</td>
<td>75.2</td>
<td>16.14</td>
<td>2.066</td>
</tr>
<tr>
<td>102</td>
<td>5</td>
<td>117.0</td>
<td>1.918</td>
<td>1.913</td>
<td>117.0</td>
<td>35.22</td>
<td>2.315</td>
</tr>
<tr>
<td>103</td>
<td>5</td>
<td>112.8</td>
<td>1.367</td>
<td>1.364</td>
<td>112.8</td>
<td>76.73</td>
<td>2.422</td>
</tr>
<tr>
<td>104</td>
<td>2</td>
<td>58.0</td>
<td>1.232</td>
<td>1.229</td>
<td>58.0</td>
<td>4.74</td>
<td>1.502</td>
</tr>
</tbody>
</table>

Table 5. ADSL2+ technology, Peers

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>100</td>
<td>16.74</td>
<td>6.994</td>
<td>1.524</td>
<td>1.384</td>
<td>6.994</td>
<td>1904</td>
<td>27.03</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>101</td>
<td>15.50</td>
<td>7.998</td>
<td>1.066</td>
<td>0.954</td>
<td>7.989</td>
<td>503.2</td>
<td>26.28</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>103</td>
<td>18.84</td>
<td>8.855</td>
<td>6.299</td>
<td>5.667</td>
<td>7.855</td>
<td>291.9</td>
<td>26.58</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>102</td>
<td>26.31</td>
<td>8.391</td>
<td>3.373</td>
<td>3.026</td>
<td>8.391</td>
<td>931.7</td>
<td>26.70</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>104</td>
<td>16.74</td>
<td>8.114</td>
<td>6.469</td>
<td>5.838</td>
<td>8.113</td>
<td>242.1</td>
<td>26.43</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>102</td>
<td>38.86</td>
<td>7.692</td>
<td>1.747</td>
<td>1.560</td>
<td>7.692</td>
<td>180.5</td>
<td>30.37</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>101</td>
<td>13.66</td>
<td>7.379</td>
<td>1.647</td>
<td>1.478</td>
<td>7.378</td>
<td>841.3</td>
<td>28.40</td>
</tr>
<tr>
<td>8</td>
<td>2</td>
<td>100</td>
<td>18.02</td>
<td>6.901</td>
<td>6.426</td>
<td>5.741</td>
<td>6.900</td>
<td>1923</td>
<td>26.16</td>
</tr>
<tr>
<td>9</td>
<td>2</td>
<td>102</td>
<td>30.21</td>
<td>7.163</td>
<td>1.214</td>
<td>1.097</td>
<td>7.163</td>
<td>199.7</td>
<td>31.24</td>
</tr>
<tr>
<td>10</td>
<td>2</td>
<td>103</td>
<td>28.97</td>
<td>8.533</td>
<td>3.310</td>
<td>2.970</td>
<td>8.532</td>
<td>294.9</td>
<td>26.42</td>
</tr>
</tbody>
</table>

Table 6. ADSL2+ technology, Rendez-Vous

<table>
<thead>
<tr>
<th>UUID</th>
<th>Links Number</th>
<th>Queue PROTOCOL Nbout [×10^3]</th>
<th>Queue PROTOCOL MResponse [×10^{-3}]</th>
<th>Queue PROTOCOL MService [×10^{-3}]</th>
<th>Queue SOCKET Nbout [×10^3]</th>
<th>Queue SOCKET MResponse</th>
<th>Queue SOCKET MService</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>5</td>
<td>50.02</td>
<td>1.114</td>
<td>1.113</td>
<td>43.66</td>
<td>72290</td>
<td>6.555</td>
</tr>
<tr>
<td>101</td>
<td>3</td>
<td>37.50</td>
<td>0.780</td>
<td>0.780</td>
<td>37.46</td>
<td>4605</td>
<td>7.409</td>
</tr>
<tr>
<td>102</td>
<td>5</td>
<td>57.27</td>
<td>1.146</td>
<td>1.145</td>
<td>53.40</td>
<td>17600</td>
<td>7.366</td>
</tr>
<tr>
<td>103</td>
<td>5</td>
<td>53.45</td>
<td>0.823</td>
<td>0.823</td>
<td>49.56</td>
<td>34490</td>
<td>7.767</td>
</tr>
<tr>
<td>104</td>
<td>2</td>
<td>31.91</td>
<td>0.896</td>
<td>0.895</td>
<td>31.91</td>
<td>40</td>
<td>5.622</td>
</tr>
</tbody>
</table>

The performance indexes reported in the Tables 1–6 refer to the DATA_MSG type of messages. They are measured in this context to value the impact that the traffic due to the VOIP communications that are activated in the P2P network has in comparison with the traffic related to the File-Sharing procedures that is characterized by having a lower priority level.
5.1.2. P2P network with file-sharing traffic

The results exposed in this section refer to the considered P2P network where the voice traffic is absent. Hence, in this case VOIP calls are not activated and the P2P network is characterized by the data traffic presence due to the activation of the file-sharing procedures which are run with the same modalities as in the previous network model. In this context the performance parameters with regard to the data traffic that is characterized by the Data_Msg object type are always measured. Due to space reasons the tables related to the two cases:

- Fast Ethernet Technology,
- Wireless Technology,

are not reported in this study.

In fact the results of index performance measurements in these two cases are equivalent to the corresponding data obtained in the presence of the VOIP traffic that are illustrated in Section 5.1.1.

Conversely, as shown in the Tables 7–8, the results are considerably different in the case that the P2P network is locally implemented with ADSL2+ technology.

**Table 7.** ADSL2+ technology, Peers

<table>
<thead>
<tr>
<th>UUID</th>
<th>GROUPID</th>
<th>Bytes Stored ([\times 10^9])</th>
<th>Queue PROTOCOL Nbout ([\times 10^3])</th>
<th>Queue PROTOCOL MResponse ([\times 10^{-3}])</th>
<th>Queue PROTOCOL MService ([\times 10^{-3}])</th>
<th>Queue SOCKET Nbout ([\times 10^3])</th>
<th>Queue SOCKET MResponse</th>
<th>Queue SOCKET MService</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>29.20</td>
<td>7.370</td>
<td>1.666</td>
<td>1.529</td>
<td>7.367</td>
<td>82.6</td>
<td>28.43</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>31.10</td>
<td>8.450</td>
<td>1.160</td>
<td>1.053</td>
<td>8.450</td>
<td>217.7</td>
<td>28.62</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>26.31</td>
<td>7.919</td>
<td>6.568</td>
<td>6.034</td>
<td>7.916</td>
<td>136.3</td>
<td>27.64</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>37.58</td>
<td>8.903</td>
<td>3.631</td>
<td>3.294</td>
<td>8.903</td>
<td>684.5</td>
<td>28.35</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>23.04</td>
<td>8.162</td>
<td>7.173</td>
<td>6.617</td>
<td>8.161</td>
<td>150.4</td>
<td>28.75</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>55.97</td>
<td>8.834</td>
<td>1.904</td>
<td>1.718</td>
<td>8.834</td>
<td>103.5</td>
<td>30.93</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>29.16</td>
<td>8.248</td>
<td>1.765</td>
<td>1.597</td>
<td>8.246</td>
<td>97.2</td>
<td>29.79</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>27.46</td>
<td>6.927</td>
<td>6.855</td>
<td>6.240</td>
<td>6.927</td>
<td>84.8</td>
<td>29.32</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>42.07</td>
<td>7.221</td>
<td>1.331</td>
<td>1.225</td>
<td>7.221</td>
<td>110.3</td>
<td>33.68</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>41.06</td>
<td>8.704</td>
<td>3.455</td>
<td>3.114</td>
<td>8.703</td>
<td>261.7</td>
<td>29.43</td>
</tr>
</tbody>
</table>

**Table 8.** ADSL2+ technology, Rendez-Vous

<table>
<thead>
<tr>
<th>UUID</th>
<th>Links Number</th>
<th>Queue PROTOCOL Nbout ([\times 10^3])</th>
<th>Queue PROTOCOL MResponse ([\times 10^{-3}])</th>
<th>Queue PROTOCOL MService ([\times 10^{-3}])</th>
<th>Queue SOCKET Nbout ([\times 10^3])</th>
<th>Queue SOCKET MResponse</th>
<th>Queue SOCKET MService</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>5</td>
<td>57.95</td>
<td>1.485</td>
<td>1.483</td>
<td>52.40</td>
<td>44930</td>
<td>9.538</td>
</tr>
<tr>
<td>101</td>
<td>3</td>
<td>40.10</td>
<td>0.859</td>
<td>0.858</td>
<td>40.09</td>
<td>133</td>
<td>8.384</td>
</tr>
<tr>
<td>102</td>
<td>5</td>
<td>61.51</td>
<td>1.362</td>
<td>1.361</td>
<td>57.81</td>
<td>25180</td>
<td>8.646</td>
</tr>
<tr>
<td>103</td>
<td>5</td>
<td>57.46</td>
<td>0.982</td>
<td>0.981</td>
<td>54.06</td>
<td>29890</td>
<td>9.245</td>
</tr>
<tr>
<td>104</td>
<td>2</td>
<td>32.16</td>
<td>0.930</td>
<td>0.930</td>
<td>32.16</td>
<td>29</td>
<td>5.990</td>
</tr>
</tbody>
</table>
6. Conclusions

If the measurements carried out on the P2P network where only the data traffic is present are compared with those carried out on the same system after adding the VOIP traffic, it can be deduced that: the VOIP traffic significantly affects the performance parameters of the P2P network with the ADSL2+ technology. In particular, the Peer SOCKET queues have a higher average response time compared with the corresponding values of the network with the data traffic only, as is illustrated in Figure 20. It can be observed that also the average service times of the Peer SOCKET queues are equivalent in the two systems, as is illustrated in Figure 21. Recalling that the response time is equal to the sum of the waiting time plus service time, this fact points out that the P2P system with VOIP traffic and ADSL2+ technologies result in congesting the DATA_MSG class customers. These customers are subject to a significant waiting time in queues inside the sockets. So we can say that in this case VOIP applications degrade the behavior of File-Sharing applications.
References

[1] Pasini L and Feliziani S 2004 TASK Quart. 8 (3) 333
[2] Pasini L and Feliziani S 2005 TASK Quart. 9 (4) 373
[3] Pasini L and Feliziani S 2006 TASK Quart. 11 (3) 203