On-line QoS estimation system for Internet accesses

Julián Fernández-Navajas, José Carlos Ibar-Cuerda, Eduardo Viruete-Navarro, Ignacio Martínez-Ruiz
Communication Technologies Group (GTC) – Aragon Institute of Engineering Research (I3A)
University of Zaragoza, Spain
{navajas,474229,eviruete, imr}@unizar.es

***Abstract*— The principal aim of this paper is to show the development of an on-line system capable of measuring bandwidth, available bandwidth and delay as Quality of Service (QoS) parameters and to do it from the point of view of the final user. The algorithm selected for this system is One-Way and is based on the transmission of known length User Datagram Protocol (UDP) packet bursts to measure packet spacing when they cross the *bottleneck* link. The system is deployed as an application which consists on a central node where users come to get their QoS measurements. This central node contains a Web server that hosts HyperText Markup Language (HTML) pages and the Java applet that implements the client-side application in the browser of the user. Another purpose of this work is to validate the developed application and to measure specific QoS parameters on the following representative technologies: Analog modem, Asymmetric Digital Subscriber Loop (ADSL), cable modem and Universal Mobile Telecommunications System (UMTS).**

I. Introduction

From its beginning, Internet has experienced a huge increase in the number of users and data transferred. The type of information transmitted has also changed and multimedia services now represent a significant amount of traffic. Access technologies have diversified to accommodate bandwidth demand, and their different characteristics lead to diverse levels of Quality of Service (QoS) [1].

When working with the common client-server architecture, QoS is determined by very heterogeneous factors as the type of server where the information resides, its geographical location, communication protocols and network technologies crossed to reach the information server. Also, QoS can be considered from different points of view: security, performance, speed, reliability, overall user impression, etc.

There are several actors involved in measuring communication QoS: network operators, Internet Service Providers (ISP) and final users. The differences between them are mainly in the measurement methods they can use and the utilization of the acquired data according to their own purposes. The network operator is interested in network planning, the ISP wants to ensure a certain degree of service for the final user, who finally wants to check the QoS obtained.

On one hand, QoS plays a crucial role in order to balance the compromise between user demands and network resources from the point of view of network operators and ISPs. On the other hand, Internet users want the best performance of their Internet accesses, which translates into the best possible QoS in every situation. Thus, these three agents can greatly benefit from a system designed to measure QoS, making it possible for them to compare what is theoretically offered with what is actually obtained from an Internet access. This is especially important for final users, given the fact that over the last years, problems derived from poor Internet accesses are in the top positions in the number of complaints to consumer associations.

QoS-related measurements can be acquired from network end points or intermediate points (edges or backbone points). Those points can be located with the aid of the general network scheme in Fig. 1. End points are illustrated as hosts, edge points as modems or routers and the cloud stands for the backbone network.

In the communication path there is usually a link that sets QoS parameters, and it is commonly called *bottleneck* [2,3]. It is frequently located in the user access, going from the traditional analog modem to the high-speed wideband digital accesses, wired or wireless. Internet access networks shown in the cloud of Fig. 1 as Digital Subscriber Loop (DSL), cable modem, analog modem over Public Switched Telephone Network (PSTN), Local Area Network (LAN) and the more recent Universal Mobile Telecommunications System (UMTS) have very heterogeneous characteristics: different bandwidth and delay, asymmetry, variable frame size, etc.

The complex set of elements that influence QoS makes its measurement a difficult task. Different estimation tools focused on discovering bandwidth in the bottleneck link employ measurement methods that can be classified into passive [4,5] and active [6,7]. The former capture traffic already present in the network and measure packet arrival times, whereas the latter use special probing packets injected into the network. Active measurement methods can be further divided into those that measure Round Trip Time (RTT) [8] and those that only measure one traffic direction (One-Way) [6]. The most used protocols in these measurement systems are User Datagram Protocol (UDP), Transmission Control Protocol (TCP) and Internet Control Message Protocol (ICMP).

Once the different measurement acquisition methods have been presented, it is very important to identify the most relevant QoS parameters. Two common and ubiquitous parameters used to measure QoS levels are bandwidth and delay [2,9-12]. Although bandwidth estimation is typically required to characterize QoS, new multimedia services (e.g., real-time audio and video transmissions) also require bounded delay and low *jitter*. In our case, we have chosen these parameters because they allow the client to check the performance of his Internet access, especially when it is used in multimedia communications. As this kind of communications mainly uses Real-Time Protocol (RTP), which in turn uses UDP, this is the protocol selected for our measurements.

Things being as they are, the aim of this paper is to show the development of a system capable of measuring bandwidth, available bandwidth and delay as the QoS parameters chosen and do it from the point of view of the final user. The algorithm selected for this system is One-Way and is based on the transmission of known length UDP packet bursts to measure its spacing when they cross the bottleneck link [3].

The rest of the article is structured as follows: in section II a teorethical review of the algorithm implemented and of the main Internet accesses is presented. Section III presents the application development. A description of the scenarios used for the tests is included in section IV. In section V, the results obtained in the tested scenarios are presented and conclusions are collected in section VI.

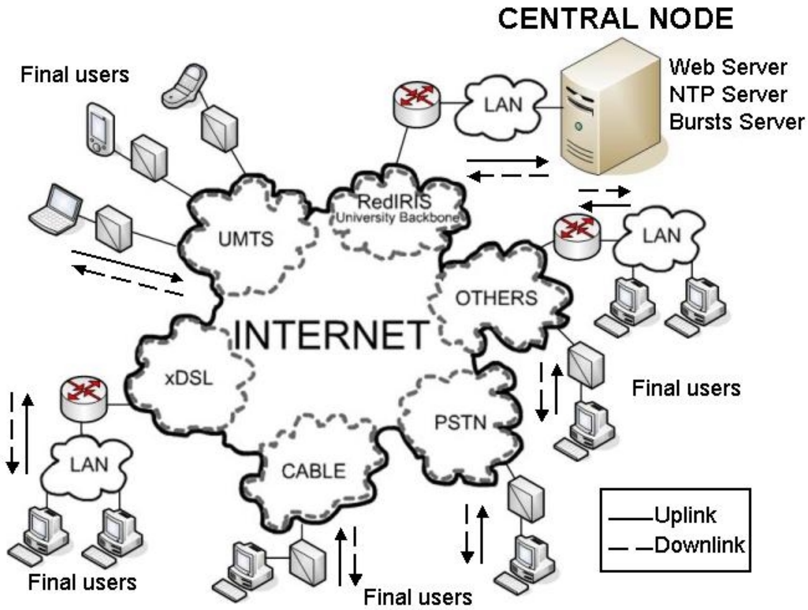

Fig. 1: Our application in a general network context.

II. Methodology

This section studies the theoretical concepts required to the right development of the system. It is structured on two different parts: the theoretical support of the algorithm used, and the review of the different access technologies and their influence in the QoS achieved.

A. Theoretical support of the algorithm

As it has been introduced above, the algorithm selected is based on the transmission of bursts of *k* UDP packets with constant packet size (*S*) and to measure its dispersion when they cross the bottleneck link [3].

First of all, we are going to define the *bottleneck bandwith* (*BBw*). Given a path between two network end points that includes *n* links L_1, L_2, … L_n with bandwidths Bw_1, Bw_2, … Bw_n, the *BBw* can be defined as [13] :

$$BBw = min\ (Bw_1, Bw_2, \ldots Bw_n) \qquad (1)$$

Next, given a link L_i with bandwidth Bw_i and traffic load TL_i, the *available bandwidth* (*ABw*) in the link is defined as:

$$ABw_i = Bw_i - TL_i \qquad (2)$$

If the packets of the burst are sent as close in time as possible, that is, with the minimum gap between them, other packets sharing the link are not likely to merge with the so closely ones in the burst. As it is explained in [2], when the burst crosses a link with less bandwidth, the packet spacing becomes higher (the packet rate becomes smaller). This increment in the packet spacing is preserved when the burst crosses higher speed links (Fig. 2), allowing us to measure the *BBw* at the reception of the burst as the sum of the packet size received in response to the burst, divided by the time between the reception of the first (t_1) and the last (t_k) packet.

$$BBw = ((k\text{-}1) \cdot S)/(t_k\text{-}t_1) \qquad (3)$$

The way to calculate the *available bottleneck bandwidth* (*ABBw)* (Fig.3) consists of sending the packets in the burst at a rate equal to the already estimated *BBw*. Now, packet spacing increase will be produced by other packets in the link merging with the burst. This increase allows the estimation of *ABBw*, by using the formula shown in (3). Moreover, the estimation of the *percentage of packet loss* (*PLRate*) can be calculated as the percentage of lost packets in the burst.

This method produces acceptable results with a minimum bandwidth waste. The parameters that characterize this algorithm (packet length, number of packets per burst, packet spacing in a burst, time between bursts) are fully configurable to select those which offer more suitable results.

An important factor to consider is the number of packets to include in the burst, as well as the size of these packets. The aim is to obtain a good estimation of the links but being as less intrusive for the network as possible. In fact, the greater the number of packets in the burst and the greater their size, the more intrusive the method. But having more packets in the burst implies more accurate estimations. Studies carried out in [6] show that, for specific environments, using five packets by burst allows to reach a tradeoff between good bandwidth estimation and little bandwidth required for the estimation method.

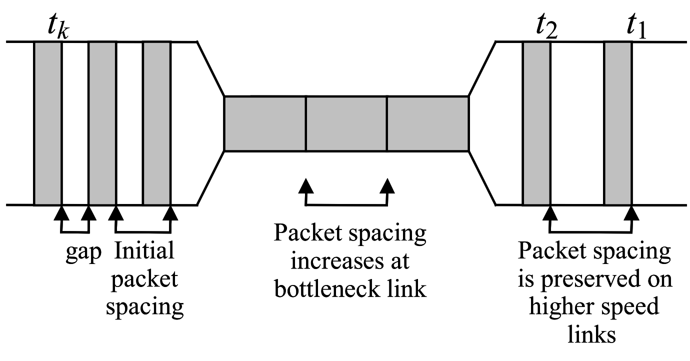

Fig. 2: Packet burst to estimate the *bottleneck bandwidth* (taken from [3])

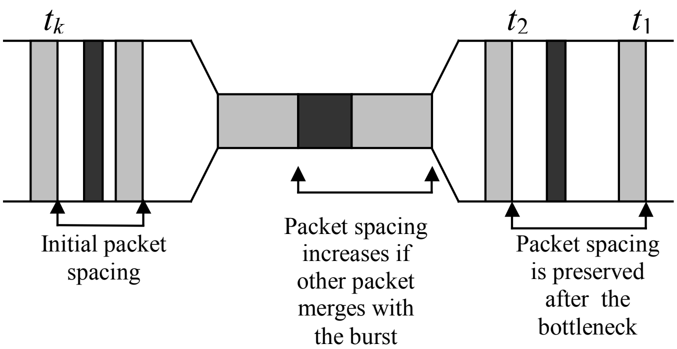

Fig. 3: Packet burst to estimate the *available bandwidth* (taken from [3])

B. Review of selected Internet Accesses

In this part we will explain the main features of several Internet access technologies used in the present work and their influence in the QoS achievement:

- *Analog modem over PSTN (Modem)*: the bandwidth of this access is low but it remains constant and 100% of its capacity is offered to the user. Frame size and flow control are configurable by the user. Depending of the modem standard, uplink and downlink can be symmetrical or asymmetrical.
- *Asymmetric Digital Subscriber Loop (ADSL)*: the capacity is greater than that of analog modem but only a percentage of it is available. Information is transmitted using fixed size Asynchronous Transfer Mode (ATM) cells [3]. As its name indicates, both capacity and available bandwidth are asymmetrical.
- *Cable modem (Cable)*: As opposed to ADSL, the access is shared between several users. Bandwidth is analogous to ADSL and the access is asymmetric too. In the uplink, Medium Access Control (MAC) is based on Time Division Multiple Access (TDMA) whereas bandwidth assignation in the downlink is controlled by the Cable Modem Termination System (CMTS).
- *UMTS*: the access is shared between some users. However, available bandwidth should remain almost constant but its value can range depending on the radio link conditions. Delay is greater than in the other accesses due to channel coding and interleaving.

III. Application Development

The application is deployed following Fig. 1. As can be seen, there is a central node where users come to get their QoS measurements. This central node contains a Web server that hosts HyperText Mark-up Language (HTML) pages and the Java applet that implements the client-side application in the browser of the user. In addition, a Network Time Protocol (NTP) server and a bursts server which takes the measures are installed in the same computer. NTP is an Internet standard protocol [14, 15] built on top of TCP/IP that assures synchronization (with millisecond accuracy) of the computer clock in a network, sending time requests to obtain server timestamps and using them to adjust the clock of the client.

Burst will be sent in both ways, first in the uplink and then in the downlink, in order to quote the QoS parameters in the bottleneck link by taking One-Way measurements.

It must be considered that, when a user connects to the central node, frames cross the Internet access network of the ISP of the user, where it is likely to be the bottleneck. So, measurements will estimate the QoS parameters of the Internet access in both uplink and downlink.

Once the system deployment has been presented, the remainder of the section will be focused on the application development. The burst sender application is programmed in Java. The main feature that makes this language well suited to be used on our measurement system is that the client-side application can be developed as an applet which can be downloaded and run directly in a Java-compatible Web browser instead of installing it. Also, Java is platform independent and allows network communications.

Applets are subject to many security restrictions that affect our application implementation: they cannot perform file I/O or make network connections except to their original host and they cannot start programs. So, the NTP and the bursts servers mentioned previously must be installed on the same computer where Web server resides.

Another important task is time precision, dependent on clock resolution, because it affects synchronization and the accuracy of parameter acquisition. Clock resolution in Java is determined by the Java Virtual Machine (JVM) implementation in each operating system and computer architecture [16]. This resolution problem increases when the packet size is low or the access rate is high because in these cases time intervals to be measured can be very small. Nevertheless, if more time resolution is needed, Java Native Interface (JNI) could be used to add C++ code [13,16], but then the resulting application will not be platform independent.

As is shown in the data flow diagram in Fig. 4, when a user loads the web page, an applet is downloaded. Once the test is executed, the applet exchanges NTP messages with the server in order to be synchronized. As soon as time offset is corrected, a connection is established via TCP with the bursts server. In fact, although the algorithm employs UDP datagrams to take the measures, TCP is used to establish parameters as the number of bursts, frames per burst, frame length, etc.

When the server processes those parameters, the bursts client is accepted or refused through the TCP connection. If the answer is affirmative, the UDP burst is sent in the uplink. To notice the end of burst, a TCP message is sent. Once the TCP message reaches the server, this one sends another UDP burst to the client to evaluate the downlink. Finally, client and server exchange their measures using TCP. This way, the results can be shown by the applet in the browser and stored by the server for further processing.

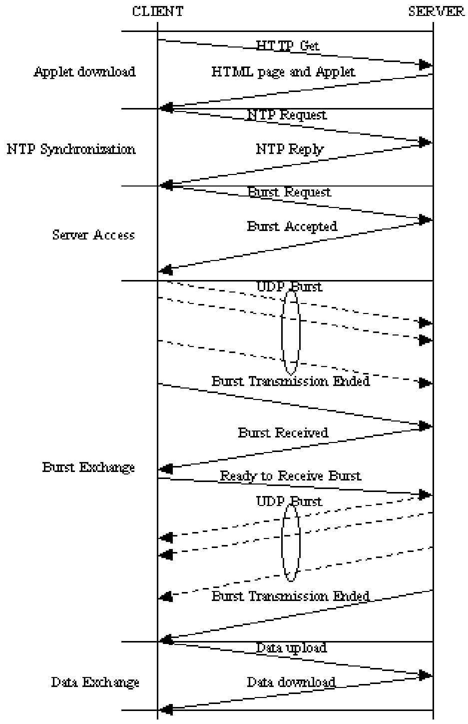

Fig. 4: Data flow diagram

IV. TEST SCENARIOS

The chosen test scenarios (shown in Fig. 5) belong to real Internet users served by commercial ISPs. It is necessary to emphasize this characteristic, since the purpose of this work is to check our developed application and to measure specific QoS parameters on the following representative technologies:

- *Modem*: V.90 standard with or without compression.
- *ADSL*: 128 kbps in the uplink and 256 kbps in the downlink, with 10% guaranteed in the contract.
- *Cable*: 128 kbps with 20% guaranteed and a 1:4 concurrence ratio.
- *UMTS*: 64 kbps in the user channel.

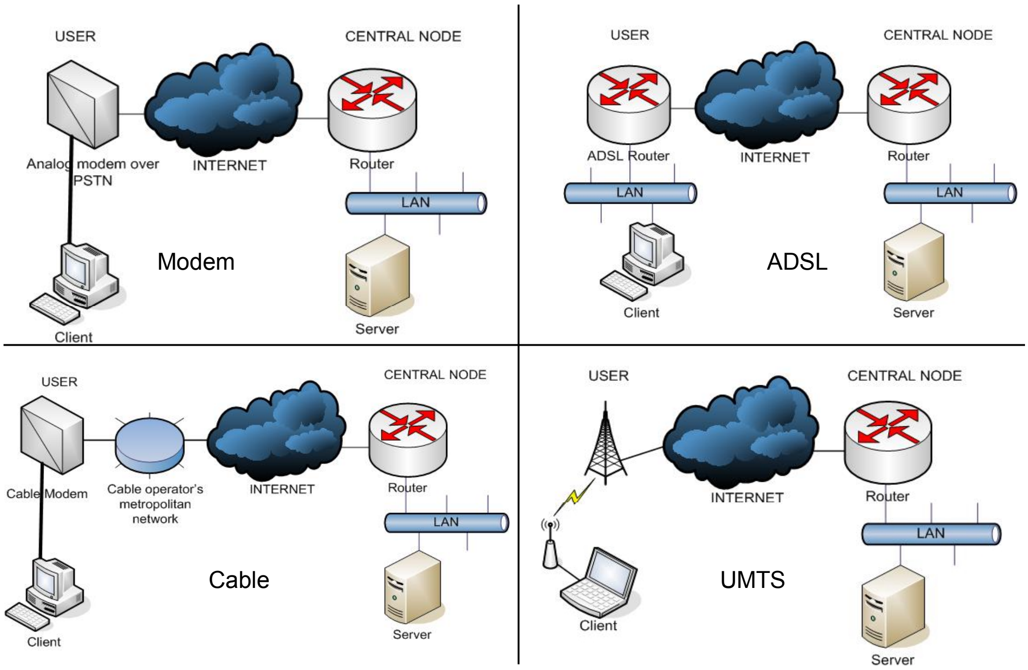

Fig. 5: Test scenarios

V. RESULTS AND DISCUSSION

The results obtained correspond to several tests consisting of:

- Number of sent bursts: 49 bursts in both uplink and downlink.
- Time between two consecutive bursts: 15 min.
- Variable frame size (*S*): 90, 120, 200, 500 bytes of UDP data without overhead.
- Frames per burst: 10 frames, sent as near as possible. This allows carrying out the study with 10 or a lower number of frames (k), by taking into account only the k firsts for the measurements. In our case, $k = 5$ and $k = 10$ have been chosen.
- Test conditions: No competing traffic, to measure *BBw* instead of *ABBw*.
- Different tests over the same link have been interleaved to concur at the same hour of the day.

TABLE I
BOTTLENECK BANDWIDTH

Access	*S* (bytes)	*k* (frames)	UPLINK		DOWNLINK	
			µ (kbps)	σ (kbps)	µ (kbps)	σ (kbps)
Modem	90	5	34.112	3.2461	47.801	10.976
		10	30.456	1.4617	46.729	6.4270
	200	5	23.810	1.3796	38.748	3.3194
		10	24.576	0.6859	40.704	1.1099
	500	5	25.602	0.9846	36.892	1.7431
		10	25.601	0.2089	38.426	1.3326
ADSL-I	90	5	23.440	1.6963	231.45	98.196
		10	23.565	0.9479	188.57	34.531
	120	5	22.153	1.1216	190.08	45.576
		10	22.248	0.6016	177.37	9.5506
	500	5	26.358	1.2772	202.78	36.384
		10	26.562	0.0833	203.01	26.252
ADSL-II	90	5	31.584	2.2856	311.86	132.31
		10	31.752	1.2772	254.09	46.529
	120	5	31.732	1.6066	272.27	65.284
		10	31.868	0.8617	254.06	13.679
	500	5	31.749	1.5384	244.25	43.826
		10	31.995	0.1003	244.53	31.621
Cable	90	5	233.36	10.299	184.57	60.485
		10	230.56	17.015	359.58	141.45
	120	5	298.39	29.214	193.78	86.228
		10	293.11	14.564	176.87	76.134
	500	5	119.95	1.7057	129.21	12.189
		10	78.421	0.3654	154.64	15.087
UMTS	90	5	57.433	13.514	61.930	7.0163
		10	59.672	10.676	61.061	2.5664
	200	5	59.433	12.973	63.083	4.685
		10	61.142	7.185	62.592	2.016
	500	5	61.387	12.441	64.537	2.9010
		10	63.246	4.9292	62.834	1.4475

Table I represents the mean (µ) and the standard deviation (σ) for the different tests in each one of the discussed scenarios:

- *Modem*: Tests were done with *S*=90, 200 and 500 bytes. As can be observed, σ decreases using all the 10 frames (instead of taking into account only the first 5), and the link is asymmetrical (with higher *BBw* in downlink). Also, the highest *BBw* was obtained by using the smallest frames (*S*=90 bytes). Moreover, if we had filled the frame with the same byte, the result could have been wrong due to the compression. So, the content of the frame must be randomized.
- *ADSL*: Tests were done with *S*=90, 120 and 500 bytes. Depending on the value of *S*, the *BBw* obtained at the IP layer (ADSL-I) may vary, this becomes even clearer in the downlink because its bandwidth is greater. It can be due to information that is transmitted physically using an entire number of ATM cells depending on *S*, and also to the low clock resolution. To check this affirmation, the way the results are calculated at the physical layer (ADSL-II) has been modified, taking into account ATM headers. This way, for *k*=10 the values of estimated *BBw* from different *S* are similar, confirming that the previously mentioned limitations have been corrected. This does not happen for *k*=5 because the clock resolution error persists. Taking all this into account, we can obtain the percentage of the contract bandwidth that the ISP is really providing us. For *S* = 500 bytes and *k*=10, the downlink reaches almost 100% but uplink is only at 20%.
- *Cable*: Tests were done with *S*=90, 120 and 500 bytes. As downlink and uplink have different systems of bandwidth

assignation, their results are treated independently. In the uplink, bandwidth estimation varies as a function of *S*. In order to clarify this result, Fig. 6 and Fig. 7 show delay between frames of each burst in reception, Fig. 6 for *S*=120 and Fig. 7 for *S*=500. In Fig. 6, delay is around 4 ms for all the frames in the burst, so the CMTS has granted all consecutive time slots. On the contrary, in Fig. 7 delay of the first three frames is 4 ms, but the rest suffer a delay of around 70 ms, which indicates that the CMTS grants the first three consecutive time slots and then it delays the next granted time slots in order to adjust user bandwidth. In the downlink, variations of estimated bandwidth also appear, but in this case σ gets higher. Thus, a detailed research would be necessary with different values of *k* and *S*.

- *UMTS*: Tests were done with *S*=90, 200 and 500 bytes. In this access, variations in the estimation originated by the encapsulation do not appear because this has been taken into account by technology to provide a 64 kbps channel, so all measurements give similar results. Furthermore, delay is much greater than in the previously mentioned wired accesses and there are more errors due to the mobile channel.

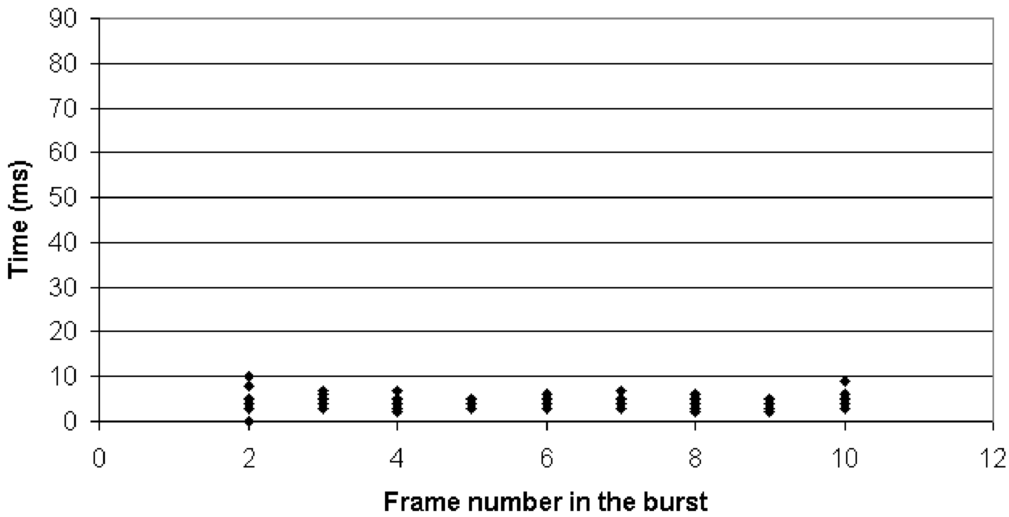

Fig. 6: Delay between frames in reception (*S*=120 bytes)

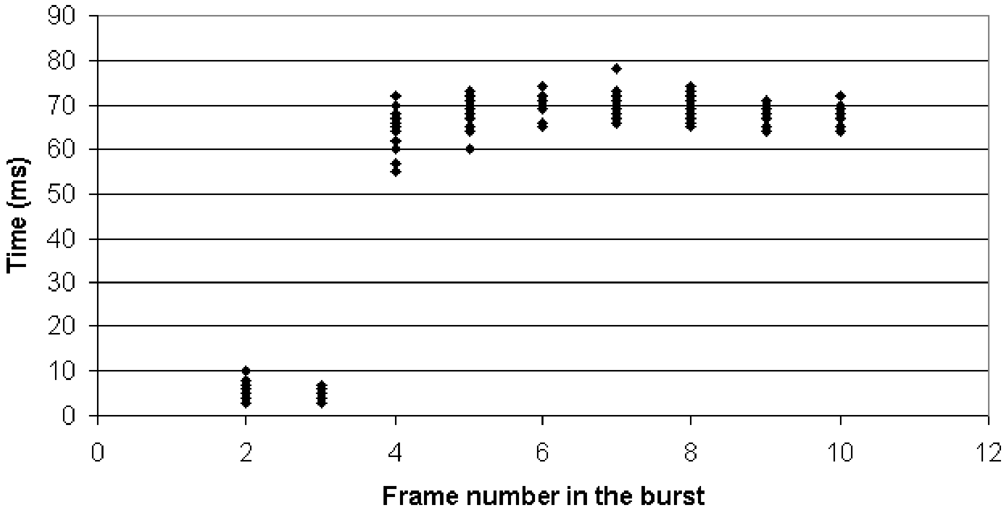

Fig. 7: Delay between frames in reception (*S*=500 bytes)

VI. Conclusions

An on-line QoS estimation system has been developed to evaluate Internet accesses. The usefulness of this system has been evidenced in the evaluation of representative accesses because ISPs offer a wide QoS range. The number of frames per burst and the frame size have a big influence on the estimated QoS. In order to obtain a suitable characterization of each access, it is very important to study the particular technology in depth. The results obtained for analog modem, ADSL and UMTS accesses are conclusive, but further research is required for cable to obtain a more complete characterization of this kind of access.

The Java-developed application has many advantages when carrying out the tests in different computers, but time accuracy and security restrictions have been problematic.

The modification of the application in order to take measurements not only between the user and the central node, but also directly between two users, and the automation of the acquired data organization for further processing, will be considered as future research lines.

Acknowledgement

This work was supported by projects TIC2002-04495-C02 and TSI2004-04940-C02-01 from Comisión Interministerial de Ciencia y Tecnología (CICYT) and European Regional Development Fund (ERDF), and project FIS G03/117 from Fondo de Investigación Sanitaria (FIS).

REFERENCES

[1] A. Vogel, B. Kerhervé, G. von Bochmann and J. Gecsei, “Distributed Multimedia and QoS: A Survey”, IEEE Multimedia, pp. 10 – 18, 1995.

[2] N. Hu and P. Steenkiste, “Evaluation and Characterization of Available Bandwidth Probing Techniques”, IEEE J. Select. Areas Commun., Vol. 21, No. 6, pp. 879-894, Aug. 2003.

[3] J. Lafuente, I. García and J. Fernández, “QoS Estimators for Client-Side Dynamic Server selection: Limitations and Keys”, The Ninth IEEE Symposium on Computers and Communications. Proc. IEEE ISCC’04 Alexandria Jun. 2004.

[4] Cooperative Association for Internet Data Analysis (CAIDA), <http://www.caida.org/tools>.

[5] NLANR Measurement and Network Analysis, <http://moat.nlanr.net>.

[6] K. Lai and M. Baker, “Nettimer: A Tool for Measuring Bottleneck Link Bandwidth”, Proc. USENIX Symposium on Internet Technologies and Systems, Mar. 2001.

[7] A. B. Downey and C. College, “Using pathchar to estimate Internet link characteristics”, Proc. SIGCOMM 1999, pp. 241-250, Sept. 1999.

[8] Z. Wang, A. Zeitoun and S. Jamin, “Challenges and Lessons Learned in Measuring Path RTT for Proximity-based Applications”, Passive and Active Measurement Workshop (PAM 2003), Apr. 2003.

[9] J. Curtis, T. McGregor, “Review of Bandwidth Estimation, Techniques”, Dept. Computer Science, University of Waikato, 2001.

[10] K. Lai and M. Baker, “Measuring Bandwidth”, Proc. IEEE INFOCOM '99, Mar. 1999.

[11] C. Dovrolis, P. Ramanathan and D. Moore, “What do packet dispersion techniques measure?”, Proc. IEEE INFOCOM '01, Apr. 2001.

[12] J. C. Bolot, “Characterizing End-to-End Packet Delay and Loss in the Internet”, Journal of High Speed Networks, Sep. 1993.

[13] Kevin T. Manley “JAVAPro: Time for Java. Use the HiResTimer class to profile the runtime of your Java code more precisely”. Available: <http://www.fawcette.com/archives/premier/mgznarch/javapro/2001/08aug01/km0108/km0108-2.asp>

[14] David Deeths, Glenn Brunette, “Using NTP to Control and Synchronize System Clocks – Part I: Introduction to NTP”, Sun Blueprints On Line – July 2001. Available: <http://www.sun.com/blueprints/0701/NTP.pdf>

[15] David L. Mills, “Network Time Protocol: Specification, Implementation and Analysis” RFC 1305 – March 1992 Available: <http://www.faqs.org/ftp/rfc/rfc1305.pdf>

[16] V. Roubtsov, “My kingdom for a good timer! Reach submillisecond timing precision in Java”. Available <http://www.javaworld.com/javaworld/javaqa/2003-01/01-qa-0110-timing.html>