Network Quality of Service Measurement System for Application Requirements Evaluation

Razvan Beuran, Mihail Ivanovici, Bob Dobinson,
CERN, 1211 Geneva 23, Switzerland, {Razvan.Beuran, Mihail.Ivanovici, Bob.Dobinson}@cern.ch
Neil Davies, Predictable Network Solutions, 36 McAfee Farm Rd, Bedford, NH 03110 USA, Neil.Davies@pnsol.com
Peter Thompson, U4EA Technologies Limited, City Point, Temple Gate, Bristol, BS1 6PL, UK, Peter.Thompson@u4eatech.com

Keywords: QoS parameters, QoS measurement, traffic monitoring, clock synchronization, performance evaluation of network applications.

Abstract

We have designed and implemented a system that permits the measurement of network Quality of Service (QoS) parameters. This system allows us to objectively evaluate the requirements of network applications for delivering user acceptable quality. We use FastEthernet taps to monitor full-duplex traffic and programmable network interface cards to extract all the information needed to compute the network QoS parameters: latency, jitter, packet loss and throughput. The measurement system makes use of a global clock to synchronize the time measurements in different points of the network.

We have employed this system to evaluate the performance of several network devices and study the behaviour of real network applications, such as file transfer and voice over IP. For these applications user perceived quality (UPQ) metrics have been defined in order to assess their QoS requirements. Since we measure simultaneously network QoS and application UPQ, we are able to correlate them. Determining application requirements has two main uses: (i) to predict UPQ for an application running over a given network based on the corresponding measured QoS parameters and understand the causes of application failure; (ii) to design/configure networks that provide the necessary conditions so that an application will run with a desired UPQ level.

1 INTRODUCTION

Quality of Service (QoS) has become more widely recognised as an important issue since network applications with real-time requirements have started to spread on a larger scale. At the moment there are not many systems able to correlate the QoS provisioned by networks with the user perceived quality (UPQ) for specific applications, such as file transfer or voice over IP (VoIP).

Knowing the requirements of such applications allows predicting whether a certain connection is valid for a certain application and what will be the perceived quality for that application.

There are various projects involved in the field of QoS. The Quantum project studied the mechanisms that can be employed for different networks in order to ensure service differentiation [7]. SEQUIN reviewed QoS issues [3] and described a QoS testbed topology [4]. TEQUILA's objective is to study, implement and validate network service definition and traffic engineering tools built upon DiffServ in order to obtain quantitative end-to-end QoS guarantees [8], [9], [22]. QBone specifies and deploys the QBone Premium Service, a virtual leased-line IP service built also on DiffServ forwarding primitives [29]. Internet2 has also started the End-to-End Performance Initiative [10], whose objective is to create a predictable and well-supported network environment. All these projects focus on network QoS mechanisms, whereas we undertook first establishing application requirements such that user expectations are fulfilled. Only subsequently can QoS mechanisms be deployed to meet these requirements.

The relationship between network conditions and application performance has been studied by a number of projects. Internet2 QoS Working Group has published a survey on QoS application needs [23]. TF-STREAM reported on best-practice guidelines for deploying real-time multimedia applications [5]. HEAnet reviewed several aspects of perceived quantitative quality of applications [25]. Generally, these approaches are qualitative – we aim to create a quantitative representation of UPQ that can be related to QoS parameters.

Several alternatives exist for measuring network QoS parameters, from simple applications, such as `ping` and `traceroute`, to more complex ones, like Iperf [30]. NetIQ Chariot, a commercial product, emulates transaction traffic from real applications and measures response time, throughput etc. The Surveyor project [28] uses Global Positioning System (GPS) to perform packet loss and one-way delay measurements (with a precision of 20 µs [19]) based on IETF IP Performance Metrics methodology. RIPE NCC Test Traffic Measurements Service is a system that monitors the connectivity of a site to other parts of the Internet [26]. We built our own system that is able to measure non-intrusively the network QoS parameters. This system is composed of commodity articles – FastEthernet taps, programmable network interface cards (NICs) – together with custom designed clock cards for time synchronization. Using these components, we have obtained a latency measurement accuracy of 1 μs, for any size packets, up to loads of 100 Mbps (see Section 3).

Note that some applications can be redesigned to perform better in current best-effort networks. For example, TCP has been designed when networks were relatively slow [18]. Nowadays networks have a much higher bandwidth, therefore new mechanisms could be used to improve performance [2]. However, our work concentrates only on standard out-of-the-box applications.

In parallel with monitoring network traffic for computing QoS parameters, we quantify the perceived quality for two applications, based on specifically defined metrics. Thus we are able to correlate network conditions with the UPQ for these applications; this allows the performance of two main tasks:

- predicting the expected UPQ for an application running over a given network taking into account the corresponding measured QoS parameters; understanding the causes of application failure by defining minimum requirements that must be met by the network;
- designing/configuring a network to provide the necessary conditions for an application to run at a desired UPQ level.

One of the major advantages of our system is its versatility. It can be used to test network devices, small local networks and even large local or wide-area networks (in this case GPS cards can be used for global time reference). We are able to measure one-way latency, which is more relevant than Round Trip Time (RTT) measurement given the usual asymmetry of networks. All our measurements are non-intrusive. After placing the taps in a network, traffic flows unaffected and we can observe the behaviour of real network applications. In addition, our system can be reprogrammed as required for future work.

Most network performance evaluation is currently done either through experimental work in real networks, simulation or an analytical approach. Emulation is a hybrid performance evaluation methodology enabling controlled experimentation with real applications. This is an integral component to our approach of studying QoS.

2 SYSTEM ARCHITECTURE

Figure 1 shows the system we have designed in a regular test setup. We use two FastEthernet taps to mirror the traffic on the link between two Linux PCs that run network applications. Traffic is fed into four programmable Alteon UTP NICs, two for each tap, in order to mirror full-duplex traffic. From each packet all the information required in order to compute the network QoS parameters is extracted and stored in the local memory as packet descriptors. The host PCs, which control the programmable NICs, periodically collect this information and store it in descriptor files. This data is then used to compute off-line the following network QoS parameters: latency and jitter, packet loss and throughput. We can calculate instantaneous or average values, and various histograms.

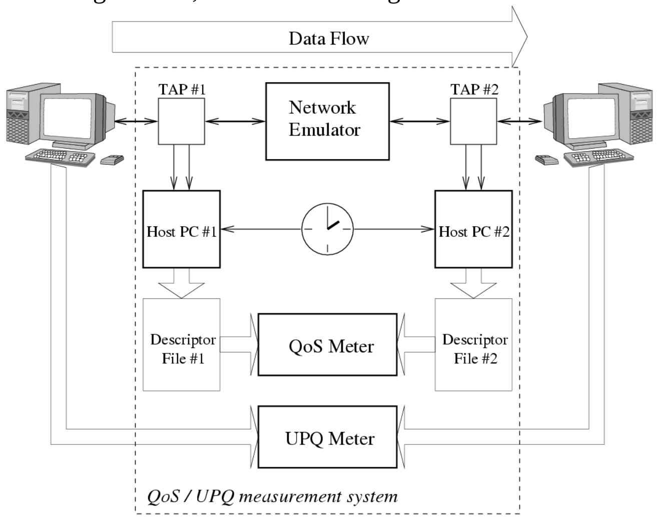

Figure 1. Measurement system setup.

An additional application-dependent process, that assesses the UPQ for that particular application, takes place during the test. For example, when studying VoIP we record the output of the VoIP system as a wave file. Then we use a specific metric, Perceptual Evaluation of Speech Quality (PESQ) score [16], to evaluate the speech quality for that VoIP communication.

The most important step is then correlating the network QoS parameters that have been computed for the connection with the UPQ calculated for the studied application. This correlation allows testing network connections before deploying network applications, and predicting the expected UPQ for those applications.

In our test setup we use a network emulator. The emulator can degrade network QoS by introducing in a controlled way artificial delay, jitter, packet loss and throughput limitations. We have used such a solution in order to be able to analyze a wide range of controllable network conditions, while using real applications. This would not have been possible using real networks or simulators.

2.1 Off-the-shelf system components

Taps are passive network devices that can be used to monitor a full-duplex link, in our case a FastEthernet connection. The setup described includes two FastEthernet taps manufactured by NetOptics that mirror the traffic flowing in both directions and feed it to the programmable

NICs (see Section 2.3). Taps are the best solution to monitor full-duplex links. Using the port mirroring option on switches has the disadvantage of introducing delays that depend on the load. Hubs are another possibility, but they do not allow full duplex monitoring due to the collisions that may appear between traffic in the two directions.

The NIST Net network emulator [24] is a freeware tool that dynamically emulates network conditions. This emulator is implemented as a kernel module that makes use of a real-time clock module in order to attain higher accuracy. The emulator allows controlled, reproducible experiments with applications that are network performance sensitive or adaptive. By operating at IP level, NIST Net can emulate the end-to-end performance characteristics imposed by various network situations.

Other components we have used are programmable Alteon Fast/Gigabit Ethernet NICs. The host PC communicates with the NIC through a shared memory segment and control structures. The NIC performs all the necessary Ethernet MAC and PHY layer processing. We programmed these NICs to monitor 100 Mbps links, operating at full speed. The NIC has a 1 MB memory, which is used to store the running software and packet descriptors extracted from the mirrored traffic.

2.2 Network QoS metrics

Based on the data collected by the QoS measurement system (see Section 2.3), we compute off-line the following QoS parameters: average latency and jitter, average throughput and packet loss [13], [6]. The average jitter is computed using a reference latency value (e.g. the latency of the first packet [13], the average latency, the latency of the previous packet [6]). The average jitter an application would experience is given by the jitter determined with respect to the latency of the previous packet. Hence we consider it the most relevant from an application-oriented perspective.

Packet loss is determined using the packet identifier from descriptors. A packet is considered lost if its identifier, which appears in the descriptor file at the first measurement point, doesn't appear in the descriptor file at the second measurement point.

We can also compute instantaneous values (e.g. for throughput) and various histograms (e.g. inter-packet arrival time histograms).

2.3 QoS measurement system

The QoS measurement system uses the information collected by NICs to compute off-line the QoS parameters for the network connection. One NIC is needed for each traffic direction, hence a total of four NICs is required in order to monitor two full-duplex links in a network. These NICs produce for each packet a descriptor with the following fields: timestamp (32 bits), packet identifier (32 bits), packet size (16 bits), protocol number (8 bits).

Timestamp represents the packet arrival time, including the time needed to store the packet in the receiving buffer. *Packet identifier* is a value that uniquely identifies the packet. We obtain it based on information from packets, such as sequence numbers from RTP and TCP headers, checksums, etc. *Packet size* contains the dimension of the packet expressed in bytes, including the four-byte CRC. *Protocol number* allows us to distinguish between different protocols and filter the packets of interest.

Synchronization between NICs is achieved by using a custom global clock system, formed of a master clock card and slave clock cards. The master clock card sends a rectangular clock signal to all slave cards. For higher robustness, the current master clock value is periodically sent to update the clock values of the slave cards. The global clock has a frequency of 66.67 MHz, derived from the 33.33 MHz PCI clock, and is used for storing system-wide valid time information. However, the NICs use processor clocks with a frequency of 88 MHz for all internal operations. Packet timestamps are obtained by transforming the local clock value to a global one, using conversion tables generated 128 times per second. The overall error of our latency measurements is determined by several factors: (i) the global clock is sourced from the PCI clock; (ii) the local 88 MHz NIC clock is obtained from a 22 MHz quartz oscillator; (iii) integer arithmetic is employed for local to global clock conversion; (iv) the NICs use DMA transfers to read the global clock from clock cards; (v) the time difference between the arrival of a packet and its processing is variable. The overall latency measurement error is bounded to 900 ns (see Section 3).

Since the clock and update signals are sent from master to slaves via short coaxial cables, this synchronization mechanism cannot be applied for remote locations. For long-haul tests, synchronization can be achieved using GPS, as described in [20].

2.4 UPQ metrics

A very important aspect of our work is the definition and quantification of application specific metrics. This allows the assessment of the UPQ for particular applications. For file transfer protocols, such as FTP or HTTP, we propose the usage of two UPQ metrics: goodput and transfer time performance.

Goodput (*G*) quantifies the network efficiency of the file transfer. It is computed as follows:

$$G = \frac{B_{\min}\,[bytes]}{B\,[bytes]}, \qquad (1)$$

where $B_{\min}$ is the minimum number of bytes required for that file transfer (including protocol overhead for Ethernet, IP, TCP and FTP) and B is the count of the actually transmitted bytes.

Goodput values are on a scale from 0 to 1, where 1 means maximum efficiency of the file transfer. Goodput decreases due to packet retransmission when packet loss occurs. *G* doesn't depend on any time parameter related to the transfer (e.g. transfer duration, RTT) but only on the amount of bytes being effectively transmitted.

Transfer time performance (*TTP*) allows the evaluation of the time efficiency for a file transfer:

$$TTP = \frac{T_{th}\ [s]}{T\ [s]} = \frac{B_{\min}[bytes]}{L\ [bps] \cdot T\ [s]}, \qquad (2)$$

where T_{th} is the theoretical transfer duration and T is the measured transfer duration. The theoretical transfer duration is the ratio of the minimum number of transmitted bytes required for that transfer, B_{min}, to the line speed, L (in our case 100 Mbps). T is computed as the difference between the time when the last packet from a transfer was received and the time when the first packet was sent.

TTP is also on a scale from 0 to 1, with 1 meaning the ideal, optimum performance. Packet retransmission delays make *TTP* values decrease. *TTP* depends indirectly on all parameters that influence transfer duration, such as RTT, TCP window size etc.

For speech quality assessment several methods exist: MOS & PAMS [14], PSQM [15], the E-model [11] and PESQ [16]. We have decided to use PESQ score, the latest metric recommended by the International Telecommunication Union (ITU). PESQ integrates the best features of PAMS and PSQM, using a perceptual model to objectively evaluate the UPQ for voice communication applications.

VoIP applications can also be studied from the point of view of network utilization efficiency, but this is generally determined by the choice of the codec and doesn't depend on network conditions. Another aspect which is however important regarding codec selection is that each of them has a maximum PESQ score that can be attained in optimum network conditions. The codec we used in our experiments is G.711 [12] that has a maximum PESQ score of 4.3.

The range of PESQ scores is from -0.5 to 4.5 [16]. A PESQ score higher than 3 means speech quality is acceptable (scores exceeding 3.8 mean toll quality, equivalent to that provided by Public Switched Telephone Networks). If scores are between 2 and 3, effort is required for understanding the speech. PESQ scores lower than 2 reflect unacceptable perceived speech quality [27].

2.4 UPQ measurement system

For file transfer experiments no extra steps are required. The corresponding UPQ parameters can be computed based on the same data used for network QoS parameters.

In order to measure VoIP UPQ the PC that receives VoIP data records the output waveform as a wave file (8 kHz sampling rate, 16 bits per sample). Resulting files are compared to the wave file used as the input of the VoIP system to calculate the corresponding PESQ score. This is done using the ITU PESQ algorithm [16] in an implementation supplied by Malden Electronics [21].

3 SYSTEM EVALUATION

Before starting our experiments we evaluated our QoS measurement system in the simplified setup from Figure 2. Since the taps are connected to each other, the time spent by a packet between the two taps is constant. We measured the difference between the arrival times at the two Host PCs. Our results show the variation displayed in Figure 3.

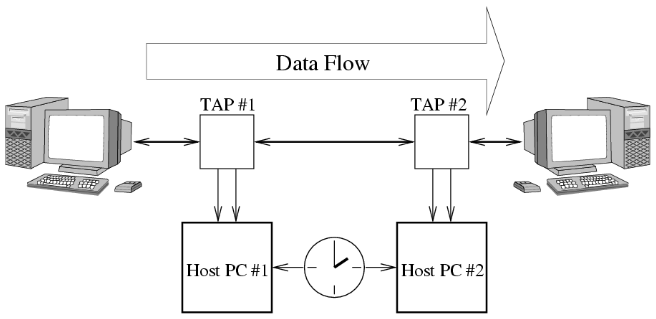

Figure 2. Setup used for evaluation.

The time difference histogram should consist of a single bin at a value equal to the propagation time between the two taps. The 900 ns spread of the histogram (400 ns full width half maximum) is small enough to allow measurement of latencies of the order of tens of microseconds. Therefore, this precision is satisfactory for our experiments.

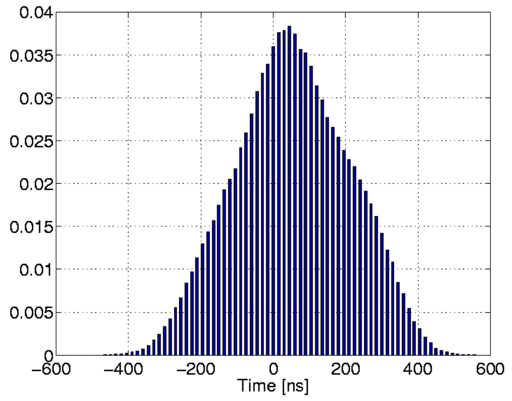

Figure 3. Histogram of arrival time differences.

We evaluated also the VoIP UPQ assessment method. The average PESQ score we obtained in zero loss, zero jitter conditions was 4.28, close to 4.3 as expected for the selected codec, G.711.

A final test determined that the network emulator could handle rates up to line speed (100 Mbps) with good precision for the artificial network degradation it introduced (packet loss and jitter).

4 EXPERIMENTAL RESULTS

We have selected two network applications that are extensively used in LANs, as well as over the Internet. They have different network behaviour:

- file transfer is an *elastic* TCP-based application. TCP tries to occupy as much of the available bandwidth as it can handle. It also adapts its transmission rate to prevailing network conditions – with high loss rates it backs off to a slower transmission rate;
- VoIP is an *inelastic* UDP-based application. UDP uses a fixed amount of bandwidth and has no inherent error recovery mechanisms so it cannot adjust to prevailing network conditions.

In the experiments we performed, we introduced artificial packet loss and jitter using the NIST Net network emulator. For file transfer tests packet loss was introduced in both traffic directions.

4.1 File transfer

We ran our tests using the setup depicted in Figure 1, with different transferred file sizes. The conditions for our file transfer tests were the following: FTP client with Linux kernel 2.4.6 (16 kB TCP window), ftp-0.17-7, FTP server with Linux kernel 2.4.9 (16 kB TCP window), wu-ftpd-2.6.1-20. In what follows, we present values obtained by averaging over 100 experiments for each intended loss rate. We ran two series of tests, one with a RTT of 0.8 ms (emulating a local network scenario) and the other with a RTT of 60 ms (emulating a wide area network). Packet loss rates ranged from 0% to 25%.

Table 1 shows the *TTP* values obtained in zero loss conditions for two different RTTs and several transferred file sizes. It can be seen that the time efficiency increases with file sizes, since the overhead of the connection establishment and termination becomes less significant compared to the file transfer time itself. The variation of *TTP* between the two RTTs is of an order of magnitude.

Table 1. Transfer time performance depending on file size and RTT.

File size		10 kB	100 kB	1MB	10MB
TTP	*0.8 ms RTT*	0.0219	0.1650	0.8221	0.8919
	60 ms RTT	0.0029	0.0141	0.0559	0.0791

The results presented below were obtained for a 10 kB file, which is the typical file size for Internet traffic [1]. For larger file sizes, the graphs of goodput and *TTP* have a similar shape. *TTP* values approach 1 for large files and small RTTs (see Table 1), which shows that it is more efficient to send the same amount of data in one large transfer than in multiple short ones.

Goodput (see Figure 4) decreases almost linearly with packet loss, showing the diminution of link utilization efficiency. As expected RTT doesn't have any influence on goodput, since *G* is not time dependent. Therefore goodput is not a stand-alone indicator of file transfer UPQ and must be correlated with *TTP*.

Transfer time performance (see Figure 5) shows the significant dependency of transfer time on packet loss. The maximum value of *TTP* equals 0.0219 due to the additional durations of connection establishment and termination, which represent approximately 96% of the transfer time for 10 kB files.

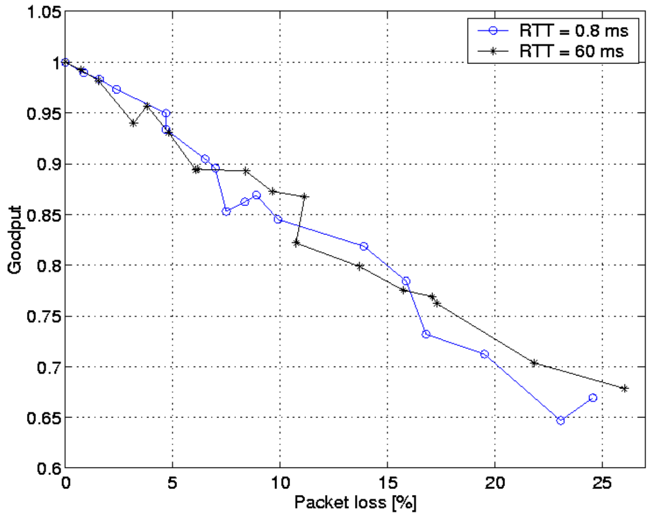

Figure 4. Goodput versus packet loss for file transfer tests (10 kB file).

Figure 5 shows that for 0.8 ms RTT, *TTP* value decreases 20 times for packet loss rates of 5% compared to the value obtained at zero loss. This is equivalent with an increase of 20 times of the transfer duration, which means a significant degradation of the UPQ. For loss rates of 10% and higher, performance degrades hundreds of times. For 60 ms RTT *TTP* is smaller than for 0.8 ms RTT and loss has a less dramatic influence on it.

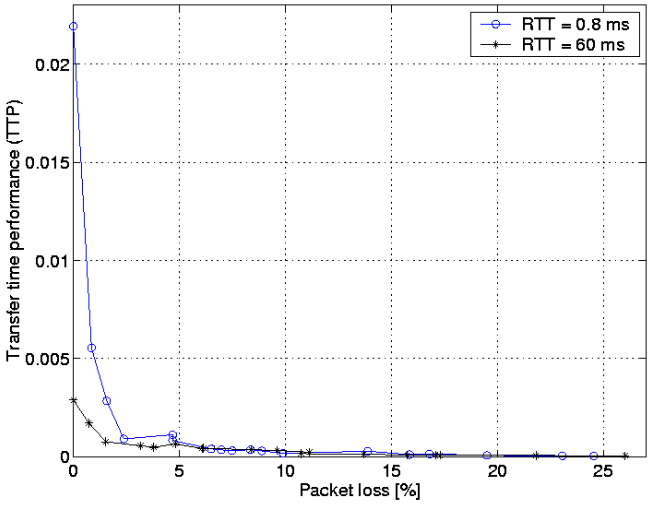

Figure 5. Transfer time performance versus packet loss for file transfer tests (10 kB file).

The influence of packet loss on TCP performance depends on the type of the lost packets: losing a data packet is easily hidden by the retransmission mechanism, whereas losing a TCP connection establishment or termination packet has a more important effect due to the relatively large timeouts. For 10 kB files, transfer duration has increased by an order of magnitude in such cases.

4.2 VoIP

VoIP is one of the most widely used interactive network applications. The bandwidth requirements of speech transmission are low (64 kbps voice data maximum), but interactivity implies high sensitivity to delay and jitter. We haven't studied the influence of one-way delay on VoIP UPQ because these requirements are generally known [17], [25]: a mouth-to-ear delay of up to 150 ms gives good interactivity, a delay between 150 and 400 ms is acceptable, and delays higher than 400 ms are unacceptable. Therefore we performed only uni-directional tests, which focus on the perceived quality of the speech itself depending on packet loss and jitter.

For our tests we used a freeware VoIP application. The results presented below are obtained with the G.711 encoding [12] (64 kbps, μ-law encoding). The application doesn't do silence suppression, reorder out-of-order packets or perform packet loss concealment. When using RTP the length of audio data per packet is of 40 ms. We used a dejittering buffer of 80 ms, that is equivalent to two VoIP packets. We present here a study of the region with loss rates between 0 and 10% and average jitter values ranging from 0 to 75 ms, since quality becomes unacceptable at these boundaries already. Five series of tests were run to collect the data used for the results shown below. Figure 6 presents a 3D plot of the PESQ score versus loss rate and average jitter. Figures 7 and 8 show cross-sections of the 3D surface along loss and jitter axes.

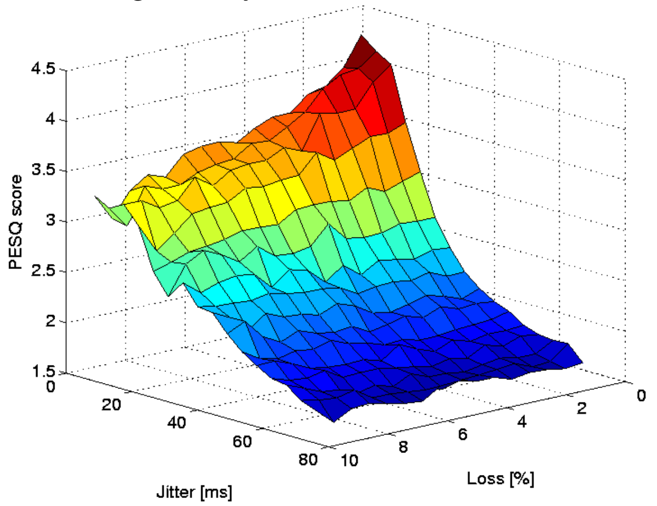

Figure 6. PESQ score versus packet loss rate and jitter for VoIP tests.

Excellent speech quality (PESQ scores ≥ 3.8) is obtained for loss rates below 2% and jitter smaller than 10 ms. Good speech quality (PESQ scores higher than 3) is obtained if jitter is less than 20 ms, even for loss rates approaching 10%.

Acceptable quality is obtained for jitter below 50 ms; for larger values the perceived quality becomes unacceptable. It is interesting to notice that for values of the jitter exceeding 25 ms the influence of packet loss on UPQ is quite small. The variation of the PESQ score is only of about 15% for loss rates ranging from 0% to 10% when jitter equals 25 ms and less than 10% for jitter values larger than 50 ms.

In some cases, for example when jitter equals 50 ms there is an increase of UPQ with loss rate (see Figure 7). We believe that this can be explained by the decrease of the perceptual effect of jitter once packet loss becomes high.

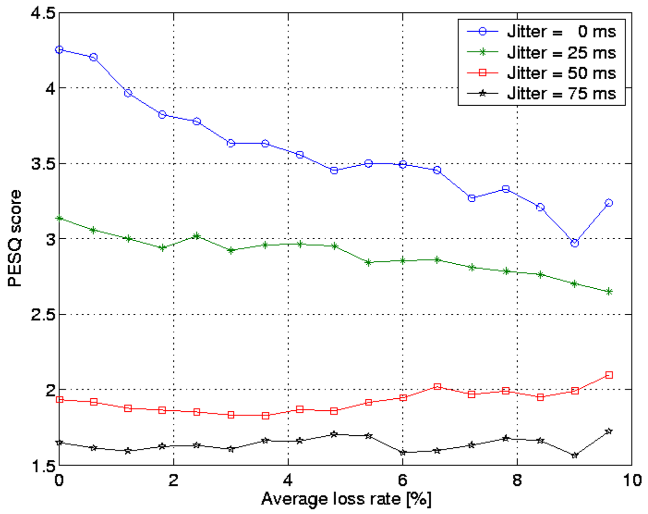

Figure 7. PESQ score versus average loss rate for VoIP tests.

The main reason for the steep decrease of quality with jitter (see Figure 8) is due to the small size of the dejittering buffer (80 ms). We have selected this size in order to have a highly interactive scenario, with an overall delay lower than 150 ms. A trade-off can be made by decreasing the required level of interactivity, thus allowing for larger dejittering buffers.

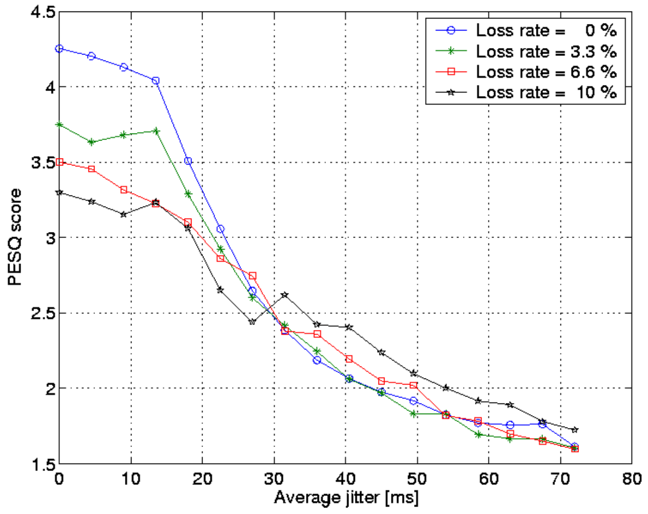

Figure 8. PESQ score versus average jitter for VoIP tests.

5 CONCLUSIONS

The novelty of our work is that we are able to both accurately measure network QoS parameters and objectively assess application UPQ in parallel. This allowed us to quantify the relationship between QoS parameters and UPQ for two applications (file transfer and VoIP) and identify their QoS requirements.

For file transfer, we observed the expected decrease of goodput with packet loss. The dependency is linear and goodput decrease is not very large in the range of 0% to 5% packet loss. For loss rates above 20%, goodput indicates a transfer efficiency lower than 0.7. The transfer time performance graph has a negative exponential shape, showing that the time needed to transfer a file increases significantly with packet loss. For loss rates around 5% and low RTTs, the *TTP* is one order of magnitude smaller than the value obtained at zero packet loss. At 25% loss rate, the time to transfer has become several hundred times larger than in the case the loss rate is smaller than 5%.

VoIP results show that for packet loss less than 10% and jitter below 20 ms the perceived speech quality is good (PESQ scores are larger than 3). For jitter exceeding 50 ms the quality of the speech signal becomes unacceptable, the distortion of the speech signal being very large.

Using our results it is possible to predict an application UPQ based on the corresponding measured network QoS parameters and understand the reasons of possible application failure. One can also determine the end-to-end network QoS requirements for an application to run with a desired UPQ level. Mapping high-level user requirements to network QoS conditions is also a key issue in Service Level Agreement contracts.

6 FUTURE WORK

In the near future we shall perform more tests with VoIP using different codecs. We also plan to quantify the influence on UPQ of dejittering buffer sizes, reordering out-of-order packets and possibly packet loss concealment.

The next step will be to study the interaction between several traffic flows sharing the same link and having the same, or different, network behaviour – elastic or inelastic. These realistic conditions will help us generalize the conclusions we have drawn so far for separate applications.

We also plan to extend our area of interest to other network applications, such as web browsing, video streaming and teleconferencing, for which UPQ is of considerable importance.

Acknowledgments

We wish to thank professor Vasile Buzuloiu for his continuous feedback and invaluable suggestions. Brian Martin, Catalin Meirosu and Jamie Lokier had a significant contribution to the early development of the clock synchronization system and pioneered Alteon NIC re-programming.

References

[1] Arlitt, M.F. and C.L. Williamson. 1996. "Web Server Workload Characterization: The Search for Invariants", Proc. SIGMETRICS, Philadelphia, PA, USA, April.

[2] California Institute of Technology. 2002. FAST Kernel for Large Data Transfers, http://netlab.caltech.edu/FAST/, November 23.

[3] Campanella, M.; P. Chivalier; A. Sevasti; N. Simar. 2001. "Quality of Service Definition", SEQUIN Project, March.

[4] Campanella, M.; M. Carboni; P. Chivalier; S. Leinen; J. Rauschenbach; R. Sabatino; N. Simar. 2002. "Definition of Quality of Service Testbed", SEQUIN Project, April.

[5] Cavalli, V. and E. Verharen. 2002. "TF-STREAM Real Time Multimedia Applications", TERENA Technical Report, March.

[6] Demichelis, C. and P. Chimento. 2002. "Instantaneous Packet Delay Variation Metric for IPPM", Internet draft, work in progress, April.

[7] Ferrari, T.; S. Leinen; J. Novak; S. Nybroe; H. Prigent; V. Reijs; R. Sabatino; R. Stoy. 2000. "Report on Results of the Quantum Test Programme", Quantum Project, June.

[8] Goderis, D. (editor). 2000. "Functional Architecture Definition and Top Level Design", TEQUILA Project, September.

[9] Griffin, D. (editor). 2000. "Selection of Simulators, Network Elements and Development Environment and Specification of Enhancements", TEQUILA Project, May.

[10] Internet2 End-to-End Performance Initiative, http://e2epi.internet2.edu/.

[11] ITU-T Recommendation G.107. 2000. "The E-model, a computational model for use in transmission planning", ITU, May.

[12] ITU-T Recommendation G.711. 1993. "Pulse Code Modulation (PCM) of Voice Frequencies", ITU.

[13] ITU-T Recommendation I.380. 1999. "Internet Protocol (IP) Data Communication Service - IP Packet Transfer and Availability Performance Parameters", ITU, February.

[14] ITU-T Recommendation P.800. 1996. "Methods for subjective determination of transmission quality", ITU, August.

[15] ITU-T Recommendation P.861. 1998. "Objective quality measurement of telephone band (300-3400Hz) speech codecs", ITU, February.

[16] ITU-T Recommendation P.862. 2001. "Perceptual evaluation of speech quality (PESQ), an objective method for end-to-end speech quality assessment of narrow-band telephone networks and codecs", ITU, February.

[17] ITU-T Recommendation Y.1541. 2001. "Network Performance Objectives for IP-Based Services", ITU, draft, October.

[18] Jacobson, V. 1988. "Congestion Avoidance and Control", ACM Computer Communication Review, SIGCOMM '88 Symposium in Stanford, CA, USA, August.

[19] Kalidindi, S. and M. Zekauskas. 1999. "Surveyor: An Infrastructure for Internet Performance Measurements", NET'99, San Jose, CA, USA, June.

[20] Korcyl, K.; G. Sladowski; R. Beuran; R. W. Dobinson; C. Meirosu; M. Ivanovici; M. L. Maia. 2003. "Network performance measurements as part of feasibility studies on moving part of the ATLAS Event Filter to off-site Institutes", First European Across Grids Conference, Santiago de Compostela, Spain, February.

[21] Malden Electronics, http://www.malden.co.uk.

[22] Manikis, D. (editor). 2001. "Overview of the TEQUILA Reference Testbeds", TEQUILA Project, February.

[23] Miras, D. 2002. "A Survey on Network QoS Needs of Advanced Internet Applications", working document, Internet2 QoS Working Group, December.

[24] National Institute of Standards and Technology, NIST Net, http://snad.ncsl.nist.gov/ itg/nistnet/.

[25] Reijs, V. "Perceived Quantitative Quality of Applications", http://www.heanet.ie/Heanet/projects/ nat_infrastruct/perceived.html.

[26] RIPE NCC Test Traffic Measurements, http://www.ripe.net/ttm/.

[27] Servis, V. 2001. "Measuring voice quality over VoIP networks", The TOLLY Group, December.

[28] Surveyor Project, Advanced Network & Services, http://www.advanced.org/surveyor/.

[29] Teitelbaum, B. (editor). 1999. "QBone Architecture", Internet2 QoS Working Group, draft, August.

[30] Tirumala, A.; F. Qin; J. Dugan; J. Ferguson; K. Gibbs. "Iperf", http://dast.nlanr.net/Projects/Iperf/.