# Measuring WCDMA and HSDPA Delay Characteristics with QoSMeT

Jarmo Prokkola, Mikko Hanski, Marko Jurvansuu and Milla Immonen
VTT, Technical Research Centre of Finland
Kaitoväylä 1, 90571 Oulu, FINLAND
Email: firstname.lastname@vtt.fi

***Abstract*—Quality of Service (QoS) is becoming increasingly important with the rise of multimedia applications (e.g., voice over IP (VoIP), video conferencing, online gaming, and Internet Protocol Television (IPTV)) in public data networks. These applications demand real-time service, while the networks should still be able to transfer also other traffic reliably (e.g., data, e-mail, WWW). We have developed a measurement tool, which is able to measure one-way QoS performance statistics. The tool enables passive monitoring of the desired application(s) and gives information about the QoS provided by the network. The tool can be used practically over any kind of network structure as long as IP is supported.**

**In this paper, we focus on detailed end-to-end delay characteristics measured in live HSDPA (High Speed Downlink Packet Access) enabled 3G network. HSDPA provides clearly better delay performance than the basic WCDMA (Wideband Code Division Multiple Access). Interestingly, deterministic long time behavior in delay was observed, and also, discrete jitter levels were found. With argumentation, and finally with a trial, we proved that these phenomena originate from the CBR-type (Constant Bit Rate) VoIP source traffic affecting with 3G network timing mechanisms and mutual clock drifts. As a side product, the study also showed that it is advisable to verify the operation of traffic generation tools before using them.**

***Keywords—passive measurements; real-time; Quality of Service; 3G; performance; VoIP***

## I. INTRODUCTION

QoS can mean different things to different people. For example, QoS is identified as network guarantees, application behavior, traffic class differentiation, and service level agreements (SLA). In addition, a distinction is made between *subjective* and *objective QoS*. The former refers to how users perceive the service level of networking applications [1]. Our focus is on objective QoS, which refers to directly measurable metrics (delay, packet loss etc.). Based on design decisions and user-experience studies, application needs can be defined in terms of these metrics [2], [3].

*Network traffic measurements* are commonly employed to study the pure network performance or the traffic itself. *Active measurements* are performed by injecting traffic with known properties into the network, while observing the network behavior and performance [4]. *Passive measurements*, on the other hand, consists of monitoring traffic at one or more points to acquire information of the traffic flow behavior (e.g., measure network load/utilization and characterize the traffic mix in terms of dominant protocols, etc), while not affecting the traffic itself [4]. Traffic traces can also be used to research and develop traffic models [5]. Both methods have their advantages, and can e.g., assist network administrators in provisioning a network, planning future expansions, and justifying resource costs [6], [7].

*QoS measurements* lie logically above network traffic measurements. Figure 1 illustrates the functional decomposition of a possible QoS measurement architecture, which includes: (a) one or more measurement points, located at network nodes (e.g., hosts, routers, firewalls); (b) traffic measurement tool, which captures packets and collects information; and (c) a QoS analysis tool, which analyzes the collected data, and finally, calculates the actual QoS statistics (d). Data analysis can be done in real-time, and/or after the traffic traces have been collected [8], [9], [10]. In addition, analysis can be centralized to a specific server, or distributed to the network.

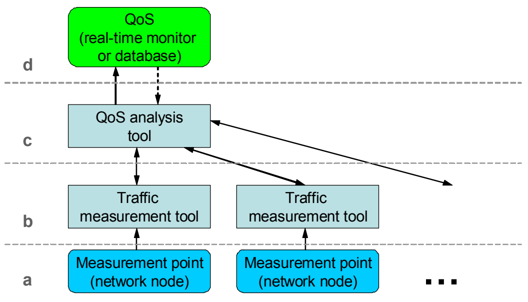
QoS
(real-time monitor or database)
d
QoS analysis tool
c
Traffic measurement tool
Traffic measurement tool
b
Measurement point (network node)
Measurement point (network node)
a
...

Figure 1. Functional decomposition of QoS measurement architecture.

In this work, we will demonstrate the developed QoSMeT tool and measure application performance over 3G network. The focus is mainly on delay behavior, which is critical for conversational applications like VoIP. We will concentrate on HSDPA and basic 3G without HSDPA (later referred to as WCDMA). Detailed information about 3G is provided in [11] and [12]. We have an end-to-end aspect from the user point of view to the performance, and thus, the measurements are done in a live 3G network including also Internet effects. The measurement scenario and majority of the studied VoIP measurement traces are the same as used in [13], which is a paper focusing mostly on the overall performance of HSDPA. Also, in [13], average statistical delay results are presented, while in this paper, we go into the very details and show several interesting findings.

## II. From Round Trip Measurements to Bi-directional Monitoring

Simple end-to-end performance measurements can be executed as single-point measurements from a terminal using some network service. Active measurement tool MOSET, for example, provides a straightforward way of measuring the performance of browsing type applications (HTTP) [14].

However, with single-point measurements, it is only possible to obtain information about the total round trip performance of the system. Often it is assumed that the links in both directions have about the same performance, but this can lead to errors, since many communication systems are asymmetric by their nature (consider e.g., cellular systems [15]). Moreover, round trip performance does not have any meaning with streaming type applications, which produce major traffic flows in one direction only. Thus, there is a need for bi-directional measurements, in which both directions can be observed independently in addition to the round trip performance (see also [16]). This can be accomplished by attaching the measurement tool to both communicating ends.

Even with this fairly simple enhancement, we come up against the basic problem of multipoint measurements: The measurement points must have some shared knowledge in order to measure the same event. In practice, this means clock synchronization, which probably is the most difficult part of it, but also *flow identification*, which might be an issue especially in a passive measurement system. By flow identification we mean that there must be a way to identify traffic flows at the measurement points, and more specifically, individual packets from the desired application.

## III. QoSMeT Architecture

A passive QoS measurement tool – QoSMeT, capable of monitoring QoS performance that a certain application experiences in the network, was developed at VTT [17]. QoSMeT enables end-to-end real-time quality measurements of any networking application(s) in both communicating directions over heterogeneous networks. QoSMeT is not only independent of transport and application layer protocols, but also link and physical layer protocols as long as they support IP. QoSMeT is also able to measure one-way delay.

In Figure 2, the structure of QoSMeT is shown. QoSMeT is a lightweight software running e.g., in the same device with the measured application. The software is installed to both communicating terminals to be able to perform bi-directional end-to-end measurements. QoSMeT is based on data link level measurements, i.e., it sniffs both the departing and arriving packets directly from the network interface device. In this way, we get close to the network itself, and are able to measure the effect of the network to the application QoS. The current version of QoSMeT is developed for MS Windows operating system, but it can be easily extended to other platforms as well.

This work was supported by Tekes (the Finnish Funding Agency for Technology and Innovation) via ITEA Easy Wireless project.

WinPcap, which is an open source packet capture library for Win32 platforms, is used for packet capturing [18]. QoSMeT records the essential packet information including accurate arrival and departure times of the packets (time stamps).

The time stamps are derived from the local clock in the terminal, but they are not added to data packets in order to keep the monitored application data flow unchanged. Nevertheless, to be able to perform QoS calculations in real-time, the exact packet information needs to be somehow exchanged between the peers during the measurement. A separate TCP connection is used to carry this control information. Naturally, this will add some overhead to the network, but in most cases, the overhead is insignificant. The user can define how often the control information is sent, so in capacity limited networks (e.g., GPRS (General Packet Radio Service)), one might choose to send it less frequently. Obviously, there is a tradeoff between the control information exchange frequency (CIEF) and the real-timeness of the monitor.

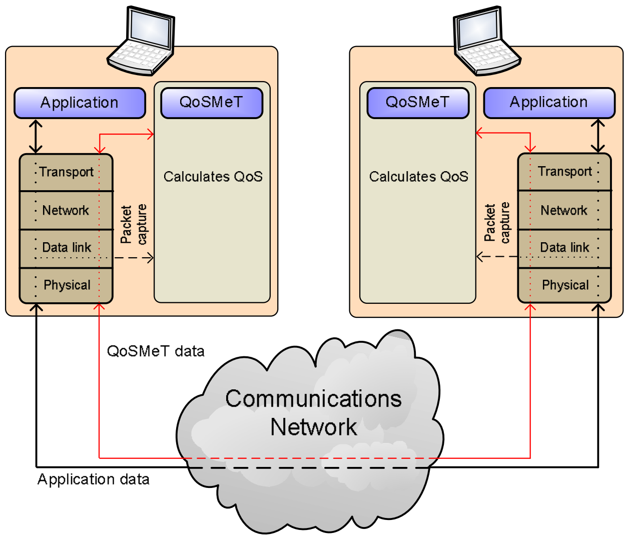

Figure 2. QoSMeT architecture.

In addition to real-time monitoring, QoSMeT supports accurate post analysis. One might choose to write not only the average values to a database, but also the exact per-packet based information. Hence, e.g., an operator is able to monitor the overall quality of the service they provide in real-time, and if something interesting happens, the accurate behavior can be examined from the database. QoSMeT can also be used in non-real-time mode, where no information is exchanged during the measurement. This mode is convenient, if only the accurate database is of interest, and/or the measured network is required to keep with minimal load. Also, in some situations, it might not even be possible to send additional control information (e.g., some cellular network operators have firewalls, preventing all atypical traffic).

### A. *Performance metrics*

QoSMeT is able to measure the following QoS metrics:

- Jitter
- Packet loss
- Connection break duration
- Offered traffic load and Throughput
- Delay

In addition, QoSMeT measures various other metrics not directly related to QoS, such as, number of received/transmitted packets, packet sizes, etc.

*Absolute jitter* ([19]) is the difference of delays between successive packets. In addition to averaged absolute jitter, for real-time monitoring, a moving average of the absolute jitter is used (as e.g., in Real-Time Protocol (RTP)) [20].

*Packet loss* is calculated by considering lost each packet that does not get an acknowledgement (corresponding arrival time stamp) by the control packet from the peer node. Even though the principle of packet loss calculation is simple, the real-time analysis makes it somewhat more complex. Since some individual packet might be delayed in the network, the calculation of packet loss must be also delayed, and in addition, CIEF should be taken into consideration.

*Connection break duration* is simply calculated as the duration of the sequential lost packets. Hence, the resolution of the break duration is dependent on the rate of the monitored traffic flow, being at worst, 1/CIEF. Yet, with real-time traffic flows the packet rate is usually high, giving good resolution to the connection break duration.

*Throughput and offered traffic load* are calculated directly from the captured packets, and there is no need to exchange this information between peers. Hence, these metrics can be calculated in real-time also in a single measurement point.

*Delay* is the most difficult performance metric. QoSMeT measures the total end-to-end delay (@ link level) experienced by individual packets, thus including all the possible delay components (e.g., propagation, queuing, and transmission delays). The precision of WinPcap time stamps using system performance counters is of the order of one microsecond, which is enough for accurate jitter calculation. However, some method is needed for the absolute timing in order to be able to calculate one-way delay. Clock synchronization is its own form of art, and is considered in several studies (e.g., [21], [22], [23]). Generally, the accuracy of PC system clocks is not adequate (it is possible that they drift even several seconds per day [6]). Commonly used network synchronization method is Network Time Protocol (NTP) [24], but the granularity provided by NTP is only about 10 ms [25], which is not enough for our purpose. GPS is a source for accurate absolute time [26]. However, PC system clock as such (especially with Windows), is not suitable for accurate time stamping even if GPS-synchronization is used [23]. Thus, in our solution, a separate GPS clock is used to acquire the time stamp directly without the system clock. We have developed our own GPS drivers to be used with WinPcap. The current implementation provides better than 100 µs absolute timing accuracy, which is well enough for this study. The calculations follow the one-way delay metric defined in RFC 2679 [16].

## IV. MEASUREMENTS

### A. Setup

QoSMeT was used to measure the delay behavior in live HSDPA enabled 3G network. The measurement scenario is presented in Figure 3. QoSMeT was installed on two laptops with Windows XP operating systems. Laptop A (Intel Pentium M 1.7 GHz processor and 1024 MB memory) was connected to VTT's laboratory LAN, which is directly connected to FUNET (Finnish university network). Laptop B (similar to Laptop A) is connected to 3G network using Option Globetrotter WCDMA and HSDPA data cards. The HSDPA card was from category 12 capable of using QPSK modulation and 5 simultaneous spreading codes.

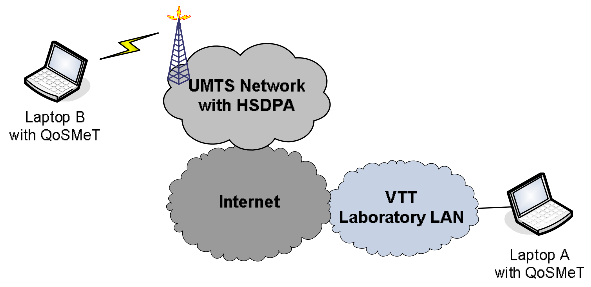

Figure 3. Measurement scenario.

A VoIP application was used for testing. Both laptops had VoIP software SJphone [27] installed on them. The software was set in a way that it will always transmit, encoding also pure noise, and hence, no actual voice conversation was needed. A Mobile IP (MIP) [28] solution was used in order to get our measurement traffic through the operator's firewall. MIP brings extra headers, thus increasing the traffic load slightly, but does not (practically) bring additional behavior to the performance, since the MIP *home agent* resides in our laboratory LAN. The traffic created by SJphone was measured with Ethereal [29] which revealed that the application packet size was 33 bytes with the headers: IP (20B), UDP (8B), MIP (4B), IP (20), UDP (8) and RTP (12B). At this point one might also wonder the efficiency of packet switched voice: 33 bytes of data vs. 72 bytes of headers + additional link/physical layer overhead. The generated traffic is CBR with approximately 50 packets per second, giving the interarrival time (IAT) of packets to be about 0.02 seconds. Therefore, the offered load is less than 50 kbit/s, being well below the capacity of the network, thus avoiding undesired queuing delays. SJphone, with our setup, does not use any traffic adaptation, etc., but rather it pushes the UDP traffic flow to the given destination IP after the connection establishment is done with SIP (Session Initiation Protocol). Hence, it is good software for measurement purposes.

### B. Results

First, we run tests without cellular network link to verify the measurement setup. The tests showed that the effects of the Internet and VTT's laboratory LAN are minor as compared to the effects of 3G network (practically, about constant, less than 20 ms delay is caused by Internet and laboratory LAN). We measured end-to-end delay behavior over HSDPA and WCDMA links at several days and different times of day during June of 2006 in Oulu, Finland (the same basic measurement set is used as in [13]). A snapshot of the per packet delay behavior in a 10 min measurement is shown in Figure 4. The difference between WCDMA and HSDPA is as expected: WCDMA downlink (DL) has a delay of mainly about 100 ms on the average with a quite high variation (jitter), while HSDPA delay is clearly steadier and lower, being on the average about 50 ms. In WCDMA, high (> 200 ms) spikes are also seen in the figure. These can quite easily be identified as retransmissions of the Acknowledgement Mode (AM) (see e.g., [30]). In WCDMA, ARQ (Automatic Repeat reQuest) mechanism is handled at RLC sublayer, and the procedure is slow, since it is operating between user terminal (UT) and RNC (Radio Network Controller). In HSDPA, HARQ (Hybrid ARQ) mechanism is used directly between the physical layers of base station (NodeB) and UT, thus allowing considerably faster retransmission procedure. The effect is clear, as HSDPA produces quite constant behavior. There are only a few high spikes, which most likely are resulting from RLC retransmissions, which are also used in HSDPA, if for some reason HARQ fails [12].

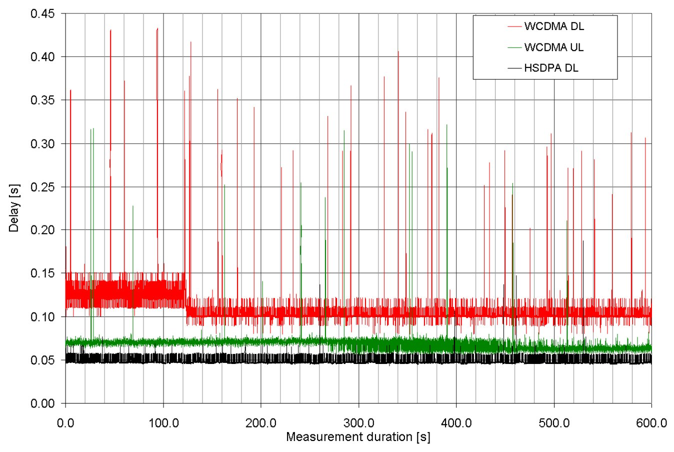

Figure 4. 3G delay measurement results (absolute values).

If we consider QoS requirements for real-time applications, 150 ms delay can be considered as a limit for good quality, while over 400 ms would be unacceptable [31]. Hence, WCDMA produces only barely acceptable quality as one-way-delay typically is above 100 ms including high spikes over 200 ms and even up to > 400 ms. Moreover, in some measurement periods, there were a considerable amount of delay spikes, decreasing thus the overall QoS seen by the user. HSDPA, on the other hand, gave quite steady results from measurement to another with average values ranging from about 45 ms to 60 ms. Spikes were also rare as is seen in our example figure, representing the typical situation. For more exact statistical information about HSDPA vs. WCDMA performance, see [13].

An interesting observation can be made in Figure 4: the WCDMA DL delay suddenly drops from average level of 130 ms to about 100 ms. In fact these two levels of delay were often found in measurements. In Figure 5, a more accurate view of the WCDMA DL delay drop section is provided. In this figure, x-axis is set to indicate the reception times of the packets, while in Figure 4, x-axis values indicated the transmission times of the packets. As seen, before the drop, delay varies a lot with about 20 ms steps, while after the drop, the delay is steadier and some 10 ms steps are found. This behavior suggests that the reason for the delay drop is originating from the changes in the radio link settings: In the beginning of the measurement, 20 ms TTI (Transmission Time Interval) is in use, but at the point of about 124 s, TTI is changed to 10 ms. A jitter of 20 ms would be a direct result from 20 ms TTI, since the radio frame length is unlikely to be a multiple of the upper layer packet. Hence, some upper layer packets will be split into two radio frames, giving extra delay (equal to TTI) for those packets. Furthermore, by examining the spikes caused by RLC retransmissions in interval ~ 127 – 129 s, it is seen that the buffered packets are sent quite rapidly after the radio link recovers. In contrast, the earlier spike (about at ~ 122 s), falls clearly slower, implying that the link speed is lower at this point. This also supports the idea that TTI is different, since with longer TTI, the link's data rate is obviously lower.

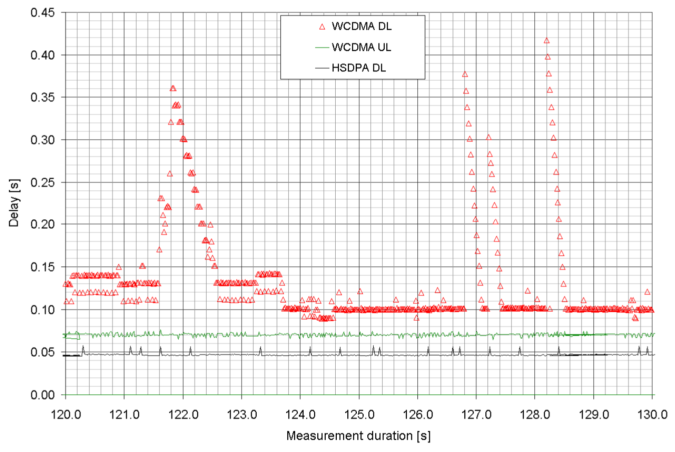

Figure 5. Zoomed 3G delay measurement results as a function of received packet time (absolute values).

By examining Figure 4 more carefully, we can see a strange behavior in WCDMA UL delay: the delay does not drop as in WCDMA DL, but rather the level seems to shift about 10 ms during the measurement. Taking a moving average of the results (not shown) verifies this observation. This interesting finding encouraged us to perform a longer continuous measurement to examine the long time delay behavior.

The results of the additional delay measurement over an 18 hour period are shown in Figure 6. The results are average values with averaging interval of 3 s, because it would not be practical to draw a single curve from over 3200000 absolute values. A very interesting observation is made: the delay behavior seems to have a triangular wave form for both, HSDPA DL and WCDMA UL (WCDMA DL gave also similar results). The first impression of this behavior made us doubt the correctness of our clock synchronization method despite the fact that it has been carefully verified to function correctly. Nonetheless, we repeated the measurement, but now without synchronization. We compensated the clock drift to reveal the delay behavior, but the result was the same (not shown).

The amplitude of the triangular wave is about 10 ms to both directions, but the wavelength is longer in DL than in UL, and also, the ramps in the DL wave seem to be the other way around when compared to UL. A similar kind of findings were also observed in [30] (even though with considerably shorter wavelengths), where it was assumed that this behavior originates from the TTI timing. We are dealing with CBR traffic with a packet IAT of 20 ms, which is a multiple of 10 ms TTI. It is very likely that a computer clock, which is tied to the traffic generation process, does not run equally with the clock in cellular network allocating the transmission times. This difference, *the mutual clock drift*, would cause the packets to arrive for transmission at different times. Thus, in the best case, a packet arrives just before the beginning of a new transmission slot, and is not affected by any additional delay, while in the worst case, a packet arrives just after a slot, and would, therefore, be delayed for TTI before transmission. A constant clock drift would therefore cause almost exactly the kind of behavior seen in Figure 6. We also verified this with a simple OPNET simulation. This explanation is also supported by the fact that the behaviors between UL and DL are different, since the clocks of the different computers run differently. Also, the direction of the drift depends on which clock runs faster (traffic generating computer clock or BS scheduling clock). We switched the UL computer to DL computer and vice versa, and the delay behaviors also switched places, while of course the average values remained the same (not shown). To back up the assumption further, we tried different computers, and got similar triangular wave type curves but with different wavelengths, as could be expected.

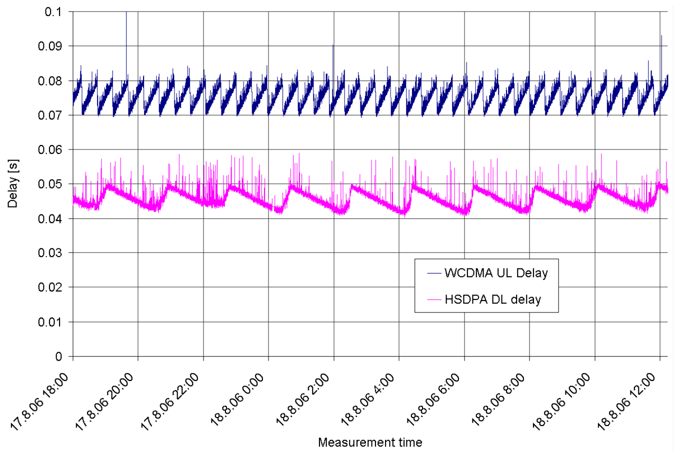

Figure 6. WCDMA and HSDPA long time average delay behavior.

The above explanation has only one pitfall: HSDPA is used in downlink and it has a TTI of 2 ms. This should result in a similar curve as in Figure 6, but with an amplitude of near 2 ms. By studying the operation of 3G radio access network, we came to the conclusion that the triangular wave behavior can be also originating from the flow control between RNC and NodeB (see e.g., [32] for more information about the $I_{ub}$ flow control), since this flow control is usually in most implementations set to have 10 ms time slots. The mechanism for getting such a delay behavior is the same as it was described within the context of TTI. Naturally, if TTI is longer than this flow control timing interval, the effect of TTI would dominate, and the triangular wave would have the amplitude near to TTI (as 20 ms amplitudes were observed in [30]). On the other hand, if TTI is shorter, as in the case of HSDPA, the effect of flow control timing dominates the behavior.

If our assumptions about the timing issues are correct, we should only see additional random variable component in the delay instead of the triangular behavior if the traffic source initiation process would be random. To test this, we first need a traffic generator, since the used VoIP software produces only CBR traffic. We had a commercial traffic generation tool available and decided to use it. It is desired that the tested random traffic process would be somewhat similar to the VoIP traffic, so we set the same constant packet length with mean IAT of 20 ms, but let the interarrival times to be exponentially distributed rather than constant. The results of the preliminary trials were quite peculiar, so we decided to test the traffic generation tool, before going into the actual measurements.

We recorded a long enough trace of the generated traffic with QoSMeT (~ 15000 packets) to reveal traffic process characteristics. From packet transmission times, we easily get IAT (assuming that the time from application to link layer is negligible). With this trial, we got the mean IAT to be about 16.1 ms with standard deviation (STD) of 23.0 ms. This was surprising, since the mean value is quite far from the desired 20 ms, and with exponential distribution, the STD should be about the same as mean. Calculating sample probability density function (PDF) and plotting it with exponential PDF revealed the characteristics of the generated traffic (Figure 7 a). As seen, the tail seems to follow quite well the exponential distribution, but there is a long gap in the distribution, and the probability mass of that gap seems to be concentrated near zero. We tried also longer traces and different mean IAT values, but with similar results. Hence, we had no choice but to try another traffic generator.

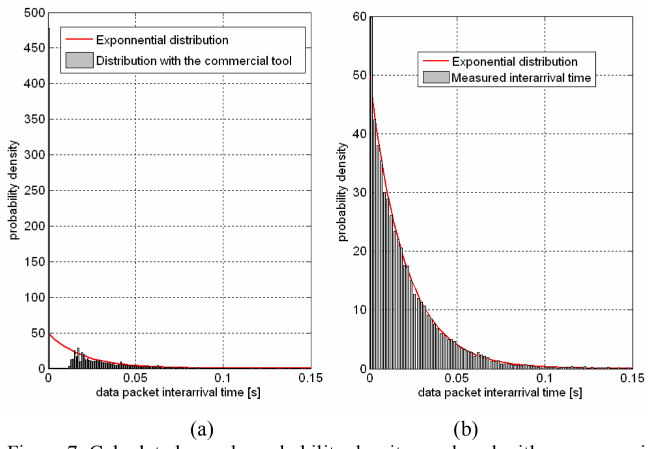

Figure 7. Calculated sample probability density produced with a commercial tool (a) and D-ITG (b) vs. exponential PDF.

Next, we tried Distributed Internet Traffic Generator (D-ITG) tool [33] with the same settings, and in the test run, we got a mean IAT of 20.0 ms and IAT STD of 21.2 ms. These values are already in the range what should be expected. Figure 7 b shows the plot of the traffic IAT PDF produced by D-ITG, and as seen, it correlates very well with the expected exponential PDF. By visual examination with *log-plot* (not shown), also the tail follows the exponential distribution well, so we do not need to go into more accurate statistical tests. Even though we have earlier found that D-ITG does not always produce precise CBR traffic, it seems to produce exponential traffic of good quality (at least good enough for this purpose).

The measurement results of the random traffic trial for WCDMA DL and WCDMA UL are seen in Figure 8. Earlier, the observed triangular wave behavior wavelength was about 30 min in UL and 110 min in DL, so if it still exists, it should be already seen in this scale of 2.5 h. However, as seen, there are not even traces of it, but instead, the delay has clearly more random behavior as compared to the CBR traffic case in Figure 6. With several trials, the level of delay variation had a lot of differences, remaining mainly higher than under CBR traffic. This changing variation is very likely caused by other traffic in the 3G network. Also, packet loss (not shown) was at somewhat higher level in this trial (e.g., in UL, the loss was now about ~ 0.12 %, while under CBR traffic, loss was typically < 0.06 %, in average). The burstiness of the exponential traffic flow can cause problems, since e.g., during the slow RLC retransmission procedure, high amounts of data might arrive for transmission, causing buffer overflows. HSDPA behavior is not shown in Figure 8, but the observations for that were the same as for WCDMA.

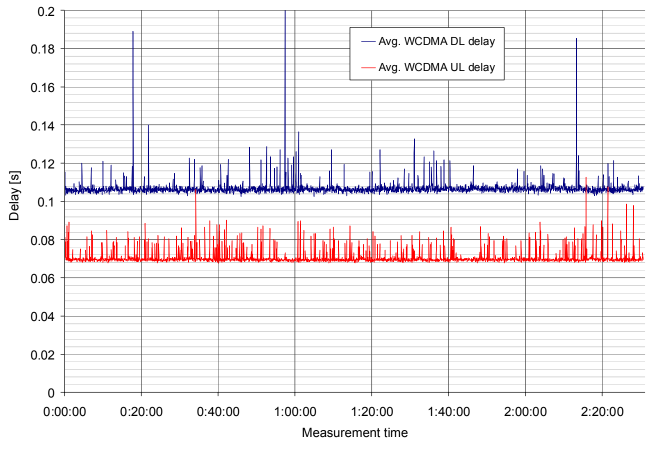

Figure 8. WCDMA long time average delay behavior with exponentially distributed source traffic.

Finally, Figure 9 shows a snapshot of the measured jitter behavior of WCDMA DL under VoIP CBR and exponentially distributed traffic loads. Quantized jitter behavior under CBR traffic is easily seen, and bands of 10 ms and 20 ms (and of course 0 ms) are clearly distinguishable. However, under exponential traffic load, jitter behavior is very random without bands, and majority of the jitter results seem to be below 10 ms (more precisely, about 55000 out of 60000 samples in this run). Interestingly, CBR type traffic also seems to cause visible ~ 1 ms discrete bands to the jitter behavior. This was also seen in other measurements excluding HSDPA DL, which only showed mainly 0, 10 ms and 20 ms bands with some random-like variation. Currently, we do have not enough information to prove the origin these 1 ms bands, but evidently they are also tied to the CBR type traffic load and its effects. Nevertheless, this trial with exponential traffic verifies our assumptions correct when explaining the interesting long time triangular wave delay behavior.

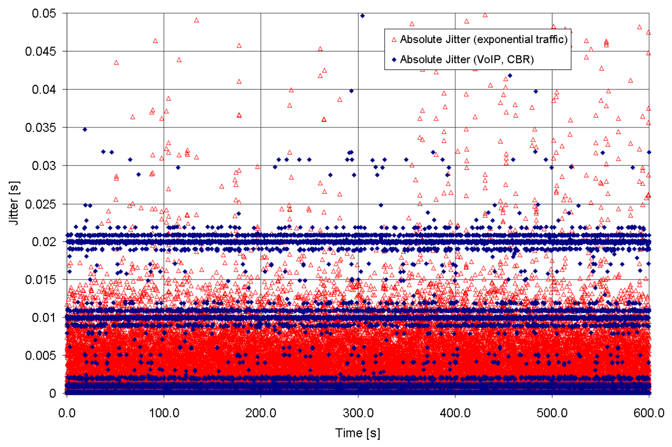

Figure 9. A snapshot of the WCDMA DL jitter behavior with CBR and exponential type traffic loads.

## V. CONCLUSIONS & SUMMARY

As QoS is becoming increasingly important, QoS measurements are also needed. Most of the available QoS tools are capable for measuring only round trip performance, but we have developed a measurement tool QoSMeT, which is able to measure one-way QoS statistics including also delay. The tool is lightweight software that can be run in normal Windows based computers. It enables passive monitoring of the desired application(s) and gives information about the QoS provided by the network. QoSMeT can be used practically over any kind of network structure supporting IP.

Interesting delay behaviors were observed in this study. With QoSMeT, we were able to verify that HSDPA provides clearly lower delays than basic WCDMA. Also, the delay variation is lower with HSDPA. HARQ provides improvements to the retransmission delays as promised, while slow RLC-level retransmissions tend to cause high delay spikes. Some effects of different TTI values were also noticed. Interestingly, a long time triangular wave behavior was observed in the results. Furthermore, jitter quantization was observed. With argumentation, and finally with a trial, we proved that these phenomena originate from the CBR-type source traffic affecting with 3G network timing mechanisms and mutual clock drifts. With random source traffic, the triangular wave behavior and quantized jitter levels do not exist. This was an interesting finding, since many real-time applications produce CBR type traffic (including the tested VoIP application). However, the observed triangular wave amplitude was only 10 ms, so it probably has meaning to the observed QoS only if the system is functioning near to the limits of acceptable performance.

As a side product, the study showed that careful attention should be put in the operation of traffic generation tools, since sometimes the generated traffic might not be even near the desired one.

Overall, HSDPA provides great improvement to delay behavior. However, the uplink is still the limiting factor and might hinder the use of two-way real-time applications like VoIP. On the other hand, even basic WCDMA is able to fulfill the average requirements for VoIP, but the observed high delay spikes can cause disturbances to the quality. It should be also kept in mind that the measurement results studied in this paper are from the network, but also the application introduces delays, making the actual user observed quality worse. Thus, real user tests should be made in order to say something accurate about the user perceived quality. Anyway, the introduction of HSUPA (High Speed Uplink Packet Access) should finally enable 3G communications for high quality real-time applications.

In this paper, the use of QoSMeT was demonstrated for evaluating accurate 3G delay behavior, and moreover, the reasons behind the particular behavior. QoSMeT is currently an end-to-end tool, but there is a need, especially for administrators and operators as well as for researchers, to be able to extract the effects of sub-networks and even network components. Hence, extending the tool being capable of handling multiple measurement points is under work. In the future, QoSMeT can be also used to provide direct input to the QoS aware applications about the network QoS conditions.

## ACKNOWLEDGMENT

We would like to thank Ali Lattunen and co. (VTT) for providing the GPS time stamping software drivers. We are graceful to Harri Holma, for his ideas and suggestions. We would also like to thank Kostas Pentikousis for his contribution to the work at the early phase.

## REFERENCES

[1] A. Bouch, A. Kuchinsky, N. Bhatti, “Quality is in the Eye of the Beholder: Meeting Users,” Requirements for Internet Quality of Service. CHI Letters, Volume 2, Issue 1, 2000. pp. 297-304.

[2] B. Somek, J. Herceg, M. Maletic, “Speech Quality Assessment,” In proceedings of Elmar '04, 2004.

[3] S. Yao, W. Lin, E. Ong, Z. Lu, “Contrast Signal-To-Noise Ratio for Image Quality Assessment,” In proceedings of ICIP '05, 2005.

[4] V. Paxson, “Towards a Framework for Defining Internet Performance Metrics”, In proceedings of INET '96, Montreal, 1996.

[5] V. Paxson, S. Floyd, ”Wide-Area Traffic: The Failure of Poisson Modeling,” IEEE/ACM Transactions on Networking, Vol. 3 No. 3, 1995. pp. 226 – 244.

[6] M. Peuhuri, “Internet traffic measurements, aims, methodology and discoveries,” Licentiate Thesis, Helsinki University of Technology, Depart. of Electrical and Communications Engineering, May 2002.

[7] C. Fraleigh, et al., “Packet-level traffic measurements from the Sprint IP backbone,” *IEEE Network*, vol. 17, 2003. pp. 6-16.

[8] K. Mochalski, J. Micheel, and S. Donnelly, “Packet Delay and Loss at the Auckland Internet Access Path,” Presented at PAM 2002, Fort Collins, Colorado, USA, 2002.

[9] P. Benko, G. Malicsko, and A. Veres, “A large-scale, passive analysis of end-to-end TCP performance over GPRS,” presented at IEEE INFOCOM 2004, Hong Kong, 2004.

[10] Y. Jiang, C-K. Tham, C-C Ko, “Providing Quality of Service Monitoring: Challenges and Approaches,” IEEE/IFIP Network Operations and Management Symposium 2000 (NOMS 2000). 10-14 April, 2000. pp.115 – 128.

[11] H. Holma, A. Toskala (eds.), ”WCDMA for UMTS,” John Wiley & Sons, Ltd. 2000.

[12] H. Holma, A. Toskala (eds.), “HSDPA/HSUPA for UMTS,” John Wiley and Sons, Ltd. first edition, 2006.

[13] M. Jurvansuu, J. Prokkola, M. Hanski, P. Perälä, ”HSDPA Performance in Live Networks,” In proceedings of IEEE International Conference on Communications (ICC 2007), 2007.

[14] M. Palola, M. Jurvansuu, J. Korva, ”Breaking Down the Mobile Service Response Time,” In proceedings of IEEE International Conference on Networks, (ICON2004), Singapore, 16-19 November, 2004.

[15] R. Lloyd-Evans, “QoS in Integrated 3G Networks,” Artech House, Inc. 2002.

[16] G. Almes, S. Kalidindi, M. Zekauskas “RFC-2679 – A One way Delay Metric for IPPM”, 9/1999, Internet RFC 2679.

[17] M. Hanski, “A Tool for Monitoring End-to-end Quality of Service in IP Networks,” Master's thesis. University of Oulu, Department of Electrical and Information Engineering, Oulu, Finland. 2005.

[18] WinPcap packet capture library, URL: http://winpcap.polito.it/.

[19] C. Demichelis, P. Chimento, “RFC 3393 – IP Packet Delay Variation Metric for IP Performance Metrics (IPPM),” 11/02, Internet RFC 3393.

[20] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson, “RTP: A Transport Protocol for Real-time Applications”, 7/03, Internet RFC 3550.

[21] C. Ellingson, R. Kulpinski, ”Dissemination of System Time,” IEEE Transactions on Communications [legacy, pre - 1988] , Volume: 21, Issue: 5 , May 1973. pp. 605 – 624.

[22] D. Mills, “Experiments in Network Clock Synchronization,” RFC957, M/A-COM Linkabit, September 1985.

[23] A. Gröhn, “Clock Synchronization of Computer Test Network,” Master’s Thesis. Helsinki University of Technology, Department of Electrical and Communications, Finland, 2004.

[24] D. Mills, "Network Time Protocol", RFC 958, 1985.

[25] V. Paxson, “Measurements and Analysis of End-to-End Internet Dynamics,” Ph.D. dissertation, University of California, Berkeley, April 1997.

[26] W. Lewandowski, J. Azoubib, W. Klepczynski, “GPS: primary tool for time transfer,” Proceedings of the IEEE, Volume: 87, Issue: 1, Jan. 1999. pp. 163 – 172.

[27] SJ Labs (16.9.2005) SJ Labs Voice over IP Software. URL: http://www.sjlabs.com/.

[28] Secgo Mobile IP software, URL: http://www.secgo.com/

[29] A. Orebaugh, “Ethereal Packet Sniffing”, Syngress Publishing, 2004.

[30] J. Cano-Garcia, E. Gonzales-Parada and E. Casiliri, “Experimental Analysis and Characterization of Packet Delay in UMTS Networks”, in Proc. of NEW2AN 2006, St. Petersburg, Russia, May/June 2006.

[31] ITU-T G.1010, “End-user multimedia QoS categories,” 11/2001.

[32] M. C. Necker, A. Weber, “Parameter Selection for HSDPA Flow Control,” in Proc. of the 2nd International Symposium on Wireless Communication Systems, Siena, 5-7 Sept. 2005, pp. 233 – 237.

[33] Distributed Internet Traffic Generator (D-ITG), URL (09.09.2007): http://www.grid.unina.it/software/ITG/.