QOMET: A Versatile WLAN Emulator

Razvan Beuran[†,*], Lan Tien Nguyen[*], Khin Thida Latt[*], Junya Nakata[*,†], Yoichi Shinoda[*,†]
[†] *National Institute of Information and Communications Technology, Ishikawa, Japan*
[*] *Japan Advanced Institute of Science and Technology, Ishikawa, Japan*
E-mail: razvan@nict.go.jp

Abstract

In this paper we present the design of QOMET, the Wireless LAN (WLAN) emulator that we develop. Our approach to WLAN emulation is a versatile two-stage scenario-driven design. In the first stage a real-world scenario representation provided by the user is converted successively into physical, data link and network layer effects that correspond to the emulated WLAN scenario. The output of the first stage is a description of the network states at successive moments of time, which is used in the second stage to accurately reproduce the wireless environment conditions by means of a wired-network emulator. We give here the details of the overall model that makes it possible to accomplish this conversion in QOMET. We then present our test methodology and illustrate our approach by several experimental results.

1. Introduction

The study of real-world WLAN environments through direct tests is made difficult by the fact that the wireless medium is difficult to control, and undesired interferences can often perturb experimental results. In addition, if mobility is to be studied, controlled movement of wireless nodes has to be orchestrated, which is a troublesome task with a high management overhead.

This is why simulation and sometimes analytical modeling are extensively used to study WLAN systems. Analytical modeling is an abstract technique that doesn't allow objective measurements, but only rough predictions of general system behavior. Simulation on the other hand, although closer to reality, is still relatively abstract, given that only models of real systems interact with each other in logical time during a simulation experiment.

A solution which gained popularity in recent years is the use of emulation for studying wireless systems. WLAN emulation combines the advantages of real-world experiments and simulation. As in the case of real-world tests, WLAN emulation allows researchers to use the same applications that are used in practice by target users. Hence, their observations are readily applicable to practical situations. Moreover, as in the case of simulation, the wireless medium effects can be controlled through emulation so as to study the desired scenarios with ease and free of interferences.

Previous approaches to WLAN emulation are however oversimplified in general. Some emulators, for instance Seawind [1] or Empower [2], introduce network layer effects, such as bandwidth limitation, delay, packet loss. However these effects are directly provided by the user who configures the emulator; this means the connection between these effects and reality is the user's task, and possibly not accurately defined. There are also attempts to develop emulators that recreate by themselves network conditions that correspond to real events. Such is the case, for example, of W-NINE [3] and the wireless-network emulation extension of SDNE [4]. Both these implementations start from a description of node positions and movements. However the accuracy of the conditions they recreate is relatively low because of the simplicity of the models used. For example W-NINE uses tables to associate IP throughput to received signal levels, loss probability is considered to be either 0 or 1, etc.

Our main research goal is the study of application performance in wireless environments, with an immediate focus on WLANs. Therefore we started developing QOMET (the acronym stands for "Quality Of applications in transforMing network Environments Testbed"), which is intended to be a versatile WLAN emulator that accurately reproduces in a wired network the WLAN conditions that correspond to a user-defined scenario. Our approach is inspired by [3] and [4] in the sense that it is a two-stage scenario-driven design. QOMET can be used either standalone, to study predefined user scenarios, or in a library form, integrated into more complex systems. Its modular architecture makes QOMET easily extendable to the emulation of other wireless environments.

The novelty of our work also appears at several levels. It consists first of all in the quality degradation view we take on networks, which is detailed in Section 2. Another aspect is the emphasis we lay on emulation realism through the use of more accurate and yet tractable models that are described in Section 3. In Section 4 we present some illustrative results obtained with QOMET that show the level of emulation detail attainable using this system. The paper ends with a section of conclusions and future work, followed by acknowledgments and references.

2. Approach overview

An overall view on our approach was previously presented in [5]. We provide here an outline. The scenario driven architecture we propose has two stages. In the first stage, from a real-world scenario representation we create a network quality degradation (ΔQ) description which corresponds to the real-world events (see Figure 1).

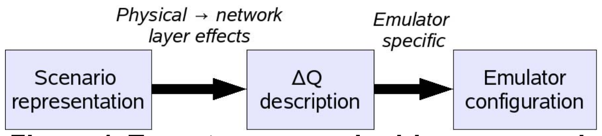

Figure 1. Two-stage scenario-driven approach to WLAN emulation

For a prior discussion of the concept of network quality degradation in the general case of computer networks see [6]. To summarize, quality degradation in networks is the change in network service quality between two measuring points. We denote this degradation by the shorthand ΔQ. An essential property of ΔQ is that the experienced degradation only increases along a network path, and cannot be undone. As a result, quality is only ever lost. A packet is either delayed or lost; a delayed packet cannot be made to arrive earlier, nor can a lost packet be recovered.

The concept of quality degradation can be applied to WLAN environments as well. Since the ΔQ description represents the varying effects of the network on application traffic, the WLAN emulator's function is to reproduce it. Hence we convert the ΔQ description calculated in the first stage into an emulator configuration that is used during the effective emulation process to replicate the user-defined scenario in a wired network. This makes it possible to study the effects of the scenario on the real applications under test. By separating the two stages it becomes possible to run QOMET on top of any wired-network emulator.

3. WLAN emulation model

This section presents the model used in QOMET to obtain the ΔQ description that corresponds to a certain scenario. The WLAN emulation model that we propose is an aggregation of several models used at the various steps of the conversion of a scenario representation to the network ΔQ description which is needed to recreate those scenario conditions. The following subsections describe the details of these models at each level of the conversion: real-world scenario to physical layer, physical layer to data link layer, and finally data link layer to network layer. Modeling stops at network layer because it is at this level that we induce the quality degradation using a wired-network emulator. For modeling we use extensively the IEEE 802.11b specification [7].

3.1. Real-world scenario to physical layer

In order to calculate the effects of real-world scenario events on the physical layer of a WLAN station, it is necessary to determine first the signal attenuation due to the distance between communicating stations, interposed obstacles, etc.

For this purpose we use the log-distance path-loss model [8]. This model gives the received power, P_r, expressed in dBm (decibel-milliwatt), as function of the received power at the distance of 1 m, P_{r0}, and the distance, d, between receiver and transmitter. The communication environment is described by the parameters α (the path-loss coefficient), σ (shadowing parameter; the standard deviation of the zero-centered Gaussian distributed random variable X_σ), and W (wall attenuation; considered equal to zero for indoor environments):

$$P_r(d) = P_{r0} - 10 \cdot \alpha \cdot \log_{10}(d) - W + X_\sigma . \qquad (1)$$

WLAN adapters have specific received power sensitivities. Based on the values provided for each adapter by manufacturers, and a negative exponential model, we can determine the frame error rate (FER) corresponding to a received power strength. This first model, the P_r-threshold-based model, computes FER_1 as function of the received power, P_r, and the adapter-specific received-power sensitivity threshold, S, as follows:

$$FER_1 = FER_S \cdot e^{\gamma(S-P_r)} , \qquad (2)$$

where γ is a constant to be determined by calibration (at the moment we use the value 1), and FER_S is the frame error rate when P_r reaches the threshold S. According to [7] and [9], FER_S equals 0.08 for 1024-byte frames.

There is another issue with WLAN adapters: their ability to discern a signal from noise. Some manufacturers, such as Intersil [10], provide the measured dependency between bit error rate (BER) and signal-to-noise ratio (SNR). We fitted an exponential function with parameters a and b on the given measured dependencies, and created a second complementary model, the SNR-based model, to determine BER_2 as function of the received-signal power, P_r, and the noise power, N (modeled as additive white Gaussian noise):

$$BER_2 = a \cdot e^{b \cdot SNR} = a \cdot e^{b \cdot (P_r - N)}. \tag{3}$$

For example, for the rate of 11 Mbps the values of parameters a and b were determined to be 12.44204 and −1.234009, respectively. In the case of the second error model we also need a frame error rate formula to determine FER_2 as function of BER_2 and the frame payload, P_{Frame} (expressed in bytes). A simplified equation that neglects frame headers, which are relatively short and transmitted at the minimum rate (hence, in principle, error-free), is:

$$FER_2 = 1 - (1 - BER_2)^{8 \cdot P_{Frame}}. \tag{4}$$

Note that both frame error rate models discussed above must be limited at 1, since their results represent probabilities.

3.2. Physical layer to data link layer

Once the BER (and FER) induced by signal propagation and interference are computed, we can pass to data link layer calculations. The two error rate models used at physical layer are produced by causes that are non-exclusive. Therefore the total frame error rate, FER, is determined using the formula for the union of non-disjoint events in probability theory, as described by the equation:

$$FER = FER_1 + FER_2 - FER_1 \cdot FER_2. \tag{5}$$

At the data link layer we also model operating rate adaptation. We propose a probability-threshold model for the ARF (Auto-Rate Fallback) mechanism [11] in order to dynamically determine the operating rate of the WLAN adapters. Given the ARF algorithm, the rate change decisions are made in QOMET according to:

$$\begin{aligned} &FER^2 > T_{down} \rightarrow down; \\ &(1 - FER)^{10} > T_{up} \wedge FER_{higher} < T_{keep} \rightarrow up, \end{aligned} \tag{6}$$

where T_{down} and T_{up} are frame error rate thresholds for taking the rate decrease and rate increase decisions, respectively. T_{keep} is used to decide whether the rate increase should really be made, depending on the frame error rate at the higher operating rate, FER_{higher}; this models the "probing transmission" feature of ARF.

The next step is to use a delay model for determining the delay, D, and the jitter, J, introduced at data link layer by the interaction between the MAC retransmission mechanism and the frame error rate. The formula we propose below computes the delay as the weighted average of the delays induced to frames undergoing a number of i retransmissions before being received, D_i, with i from 0 to r, where r is the maximum number of retransmissions (in addition to the initial first transmission of a frame). Default values for r are 6 and 3, depending whether the RTS/CTS (Request To Send/Clear To Send) mechanism in IEEE 802.11 MAC is disabled, or enabled, respectively [7].

$$\begin{aligned} &D = \frac{\sum_{i=0}^{r} (1 - FER) \cdot FER^i \cdot D_i}{\sum_{i=0}^{r} (1 - FER) \cdot FER^i}, \\ &\qquad FER \neq 0 \wedge FER \neq 1; \\ &D\big|_{FER=0} = D_0; \quad D\big|_{FER=1} = \infty. \end{aligned} \tag{7}$$

The weights included in Equation (7) represent the probabilities for a frame to undergo i retransmissions. Delay values, D_i, are computed according to the equation below, where T_{SIFS}, T_{ACK}, T_{DIFS}, T_{Frame} represent the time needed to transmit a SIFS (Short Inter Frame Space) frame, an acknowledgement frame, a DIFS (Distributed coordination function Inter Frame Space) frame, and the frame payload itself, respectively.

$$\begin{aligned} D_0 &= T_{SIFS} + T_{ACK} + T_{DIFS} + \\ &\quad T_{Backoff\_0} + T_{Frame}; \\ D_i &= D_{i-1} + T_{SIFS} + T_{ACK} + T_{DIFS} + \\ &\quad T_{Backoff\_i} + T_{Frame}, \quad i = \overline{1, r}. \end{aligned} \tag{8}$$

Note that in case the RTS/CTS mechanism is enabled, additional terms must be considered, namely T_{RTS} and T_{CTS}, which represent the time needed to transmit an RTS and a CTS frame, respectively, as well as twice more T_{SIFS}. PCF (Point Coordinate Function) 802.11 operation was not considered in our model.

In Equation (8) $T_{Backoff\_i}$ represents the average waiting time induced by the 802.11 back-off mechanism in the case of i retransmissions. The value of $T_{Backoff\_i}$ depends on the size of the congestion window after i retransmissions, CW_i, and for its computation the interval [0, CW_i−1] is uniformly sampled. The average back-off time is therefore equal to $D_{slot} \cdot CW_i/2$, where D_{slot} represents the duration of a congestion window slot in the IEEE 802.11 MAC, and is equal to 20 µs.

Jitter is computed using a similar weighted average formula with that for delay:

$$J = \frac{\sum_{i=0}^{r}(1-FER)\cdot FER^{i}\cdot J_{i}}{\sum_{i=0}^{r}(1-FER)\cdot FER^{i}}, \qquad (9)$$
$$FER \neq 0 \wedge FER \neq 1;$$
$$J\big|_{FER=0} = D_{slot}\cdot\frac{CW_{0}+1}{4}; \quad J\big|_{FER=1} = \infty.$$

The jitter values for frames undergoing a number of i retransmissions before being received, J_i, can be quickly computed using the following equation:

$$J_{i} = \left|D_{i} - D\right|, i = \overline{0,r}, FER \neq 0 \wedge FER \neq 1 \qquad (10)$$

3.3. Data link layer to network layer

After data link layer parameters are computed, we can proceed to the last step: calculating the network layer parameters. These parameters are the output of the first stage of the emulator, and they can be used to configure a wired-network emulator so as to reproduce the WLAN conditions associated to a given scenario.

Packet loss rate, *PLR*, is computed from *FER* by taking into account the 802.11 MAC retransmission mechanism:

$$PLR = FER^{r+1}. \qquad (11)$$

Delay and jitter at network layer will be those calculated in Section 3.2. A constant value can be added to that given by Equation (7) if one wishes to account for delays that are independent on the WLAN operation itself, such as packet processing delays, etc.

The other important parameter at network layer is bandwidth. The bandwidth model we propose to determine the effectively available bandwidth as "perceived" at network layer is given by the equation:

$$B = \frac{T_{Frame}}{D}\cdot R, \qquad (12)$$

where R is the current operating rate of the WLAN station.

Note that the formulas given in the previous sections for the computation of delay and bandwidth only take into account the environment effects on communication. However, if multiple users share the wireless media, additional quality degradation occurs through the use of the CSMA/CA (Collision Sense Multiple Access with Collision Avoidance) mechanism of 802.11. To account for these effects we propose the following formula to compute the average delay when the wireless medium is shared by n users, D_n, which is based on the analytical model given in [12]:

$$D_{n} = D\cdot\sqrt{n\cdot\log_{2} n}. \qquad (13)$$

To compute the corresponding available bandwidth when the wireless medium is shared by n users, B_n, one should replace D with D_n in Equation (12).

4. Experimental Results

This section shows experimental results that illustrate our approach to WLAN emulation and the capabilities of QOMET. Consider a scenario with an 802.11b node that from an initial distance of 10 m with respect to another station moves linearly on a perpendicular direction with a speed of 0.5 m/s for a duration of 30 s, and then returns with the same speed to the initial position. This is a basic scenario fragment representative of a user moving in a building while making a VoIP over WLAN phone call to another user. The selected application for this experiment was VoIP since it is probably the most used application in mobile wireless networks. The simplified motion pattern was only chosen for the sake of clarity.

The environment was modeled with $\alpha = 5.6$ (difficult reception conditions), and P_{r0} was considered −20 dBm. Parameters σ and W were assumed to be equal to zero, again for simplicity reasons. Using Equation (1) we computed the received power, P_r.

Equations (2)-(5) and (11) are then used to compute the packet loss rate. A standard noise level of −100 dBm was considered, and P_{Frame} was equal to 400 bytes, which approximates VoIP packets as sent by our VoIP system (see below). RTS/CTS mechanism was considered disabled. Subsequently, Equations (6) and (12) were used to compute the operating rate and maximum available bandwidth, respectively. The results are displayed in Figure 2.

Following that, Equations (7)-(10) are used to compute the delay and jitter for the packets communicated by the two nodes; their variation versus time is shown in Figure 3.

Once all these parameters are computed, we can proceed to the effective emulation process. In principle, the above computations can also be done in real time, during effective emulation, so that the two stages of our approach are finely interleaved. This can be useful when emulating robot systems in which motion is determined based on the communication that takes place between robots. We have separated the two stages in this explanation only for clarity reasons.

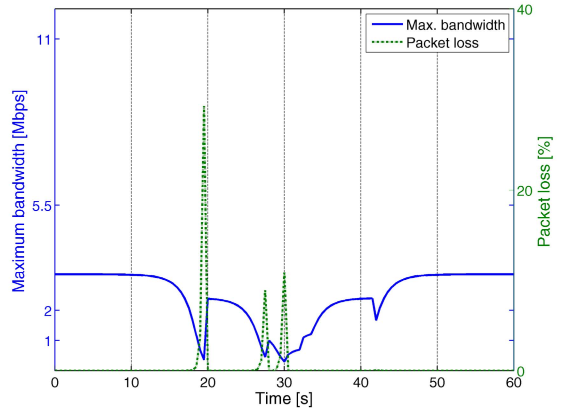

Figure 2. Maximum available bandwidth and packet loss rate versus time

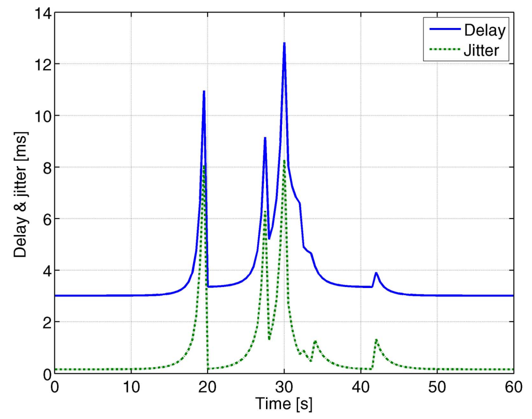

Figure 3. Average delay and jitter versus time

The experimental setup we used for the second stage, the effective emulation, is shown in Figure 4. The emulated WLAN was run on a FreeBSD PC with *dummynet* support [13]. We also made possible the execution of QOMET on StarBED, which is the large scale network-experiment environment of the NICT Hokuriku Research Center in Ishikawa, Japan [14]. Any other wired-network emulation platform could be used as well. The end nodes in Figure 4 are usual PCs that represent the mobile nodes in our emulated environment. We can compute the QoS parameters of the emulated WLAN by means of the "QoS Meter" block that uses sniffed traffic traces. Simultaneously we measure application-level performance with the "UPQ Meter", by employing application-specific performance metrics (UPQ stands for "User-Perceived Quality").

For VoIP we used first the ITU-T E-model [15] to predict voice communication quality based solely on ΔQ descriptors. The output of the E-model is the R-value, which is on a scale from 0 to 100. The next step was to use the same ΔQ descriptors to drive the *dummynet* network emulator while real voice data was sent through the network. We used a customized version of the SpeakFreely 7.6a application [16] that we modified to save the output voice signal. For this test SpeakFreely was configured to make use of the codec G.711 [17]. The voice input consists of standard voice test files supplied with the ITU-T P.862 recommendation [18]. In order to estimate the VoIP quality for real voice data we used the PESQ score, standardized by the ITU-T P.862 recommendation, which requires both the input and output voice signals.

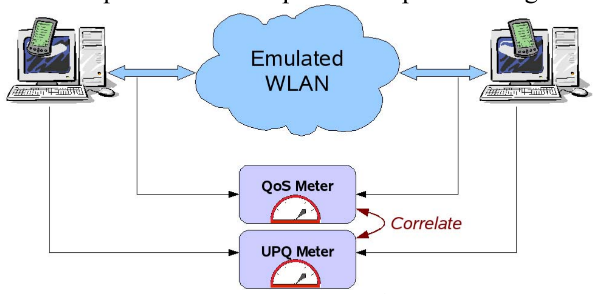

Figure 4. WLAN emulation for application performance assessment

The PESQ score and the R-value can be converted to the Mean Opinion Score (MOS) scale [19] using methods provided in the respective ITU-T recommendations. This makes it easy to compare results obtained using the two metrics, as represented in Figure 5. A value of 4.5 on the MOS scale indicates optimum quality, with good quality being associated to scores higher than 3. Quality is considered acceptable for scores between 2 and 3, whereas scores lower than 2 indicate unacceptable quality.

We can observe in Figure 5 that the MOS R-value and the PESQ score give similar results, which is a confirmation of our emulation procedure. For non-ideal conditions, around moments of time 19 s, 27 s, and 29 s, the PESQ score shows, as expected, a more realistic and lower quality level.

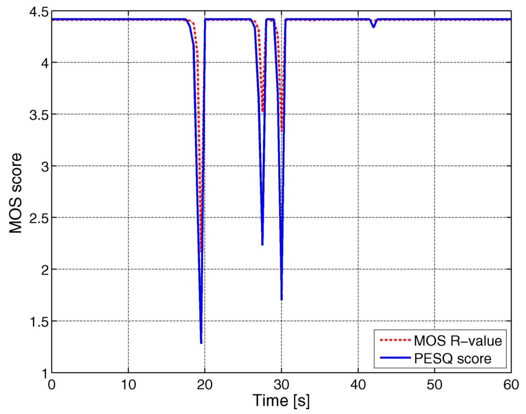

Figure 5. VoIP performance measured using the MOS R-value and PESQ score in the emulated WLAN scenario

In Figure 5 we see that VoIP quality degradation is related to operating rate changes, preceding them by several seconds. Rate changes are precisely some of the aspects simplistically modeled in other WLAN emulation approaches. Since rate changes are frequent for real motion patterns and shading conditions, their effects must to be considered when performing WLAN emulation. Another significant quality drop occurs when the mobile node is about to go out of the communication range (at time t = 30 s); then quality improves as the node starts moving back.

5. Conclusions

The approach we propose for WLAN emulation, and that we implemented in QOMET, allows the transformation of a user-meaningful real-world representation of a WLAN environment (the "scenario representation") into a network quality degradation description (the "ΔQ description"). This description is sufficient to subsequently configure a wired-network emulator and effectively reproduce an environment that corresponds accurately at network level to the emulated WLAN scenario.

We illustrated the practical use of our approach through a simple real-world scenario, for which we determined the induced network quality degradation. We then recreated the quality degradation in a wired network on which we ran the application under test, VoIP. We quantified the influence of the quality degradation on VoIP User-Perceived Quality in an objective manner using two ITU-T recommendations concerning expected and predicted user satisfaction for VoIP communication.

One aspect of our future research is to use QOMET to investigate application performance assurance. We intend to use the same setup to study the issue of performance guarantees in WLAN environments, which is vital in emergency conditions, such as disaster rescue operations and other critical situations. Another aspect of future work includes, of course, continuing the development of QOMET. A few of the features that would increase its usability are: definition of scenarios in a more realistic way (including streets and buildings), routing, etc.

6. Acknowledgments

We would like to acknowledge the contribution to the ideas in this paper of Toshiyuki Miyachi, Takashi Okada, Saber Zrelli, Assist. Prof. Satoshi Uda, Assist. Prof. Ken-ichi Chinen, and Assoc. Prof. Yasuo Tan.

7. References

[1] M. Kojo, A. Gurtov, J. Manner, P. Sarolahti, T. Alanko, K. Raatikainen, "Seawind: a Wireless Network Emulator", *Proc. of MMB 2001*, Aachen, Germany, September 2001.
[2] P. Zheng, L.M. Ni, "EMPOWER: A Network Emulator for Wireline and Wireless Networks", *Proc. Of IEEE Infocom 2003*, San Francisco, U.S.A, April 2003.
[3] T. Perennou, E. Conchon, L. Dairaine, M. Diazet, "Two-Stage Wireless Network Emulation", *Proc. of WCC2004*, Toulouse, France, August 2004.
[4] M. Bateman, C. Allison, and A. Ruddle, "A Scenario Driven Emulator for Wireless, Fixed and Ad Hoc networks", *Proc. of PGNet2003*, Liverpool, U.K., June 2003, pp. 273-278.
[5] R. Beuran, K. Chinen, K.T. Latt, T. Miyachi, J. Nakata, L.T. Nguyen, Y. Shinoda, Y. Tan, S. Uda, S. Zrelli, "WLAN Emulation on StarBED", *Proc. of ICWMMN 2006*, Hangzhou, China, November 6-9, 2006, pp. 856-859.
[6] R. Beuran, M. Ivanovici, N. Davies, B. Dobinson, "Evaluation of the Delivery QoS Characteristics of Gigabit Ethernet Switches", *CERN-OPEN-2005-002*, CERN, Geneva, Switzerland, December 2004.
[7] ANSI/IEEE Standard 802.11, 1999 Edition (Reaffirmed 2003), *http://standards.ieee.org/getieee802/download/ 802.11-1999.pdf*.
[8] T.S. Rappaport, "Wireless Communications: Principles and Practice", *Prentice Hall PTR*, 2nd edition, 2002.
[9] Intersil, "Measurement of WLAN Receiver Sensitivity", *Intersil technical brief*, February 2000.
[10] Intersil, "HFA3861B: Direct Sequence Spread Spectrum Baseband Processor", *Intersil data sheet,* February 2002.
[11] A. Kamerman, L. Monteban, "WaveLAN-II: A high-performance wireless LAN for the unlicensed band", *Bell Lab Technical Journal*, 1997, pp. 118-133.
[12] P. Gupta, P.R. Kumar, "Capacity of wireless networks", *technical report*, University of Illinois, U.S.A, 1999.
[13] L. Rizzo, "Dummynet FreeBSD network emulator", *http://info.iet.unipi.it/~luigi/ip_dummynet*.
[14] T. Miyachi, K. Chinen, Y. Shinoda, "StarBED and SpringOS: Large-scale General Purpose Network Testbed and Supporting Software", *Proc. of VALUETOOLS'06*, Pisa, Italy, October 11-13, 2006.
[15] ITU-T, "The E-model, a computational model for use in transmission planning", *ITU-T Recommendation G.107*, March 2005.
[16] B.C. Wiles, J. Walker, "Speak Freely 7.6a", *http://www.speakfreely.org*.
[17] ITU-T, "Pulse Code Modulation (PCM) of voice frequencies", *ITU-T Recommendation G.711*, 1993.
[18] ITU-T, "Perceptual evaluation of speech quality (PESQ), an objective method for end to end speech quality assessment of narrow-band telephone networks and codecs", *ITU-T Recommendation P.862*, February 2001.
[19] ITU-T, "Methods for subjective determination of transmission quality", *ITU-T Recommendation P.800*, August 1996.