Providing QoS Through Machine-Learning-Driven Adaptive Multimedia Applications

Pedro M. Ruiz, Juan A. Botía, and Antonio Gómez-Skarmeta

***Abstract*—We investigate the optimization of the quality of service (QoS) offered by real-time multimedia adaptive applications through machine learning algorithms. These applications are able to adapt in real time their internal settings (i.e., video sizes, audio and video codecs, among others) to the unpredictably changing capacity of the network. Traditional adaptive applications just select a set of settings to consume less than the available bandwidth. We propose a novel approach in which the selected set of settings is the one which offers a better user-perceived QoS among all those combinations which satisfy the bandwidth restrictions. We use a genetic algorithm to decide when to trigger the adaptation process depending on the network conditions (i.e., loss-rate, jitter, etc.). Additionally, the selection of the new set of settings is done according to a set of rules which model the user-perceived QoS. These rules are learned using the SLIPPER rule induction algorithm over a set of examples extracted from scores provided by real users. We will demonstrate that the proposed approach guarantees a good user-perceived QoS even when the network conditions are constantly changing.**

***Index Terms*—Author, please supply your own keywords or send a blank e-mail to keywords@ieee.org to receive a list of suggested keywords..**

I. Introduction

THE UTILIZATION of real-time adaptive applications for internetworking multimedia over future wireless and mobile networks is something already agreed in the wireless research community. These networks are formed by a variety of heterogeneous wireless technologies causing an unpredictably capacity-changing network scenario. In these scenarios, the network-layer quality of service (QoS) mechanisms cannot guarantee a stable service because most of the variability is due to the wireless channel itself. Adaptive multimedia applications, being able to dynamically change their settings to adapt to the available network resources can alleviate the problems caused by such unpredictable variations in the network conditions.

The way in which QoS-aware adaptive applications in the literature auto-configure themselves is commonly driven by low-level QoS parameters such as the bandwidth, jitter, delays, packet losses, etc. Their objective is finding a set of settings (audio and video codecs, video sizes, etc.) reducing the data rates to those that the network can support. However, in the author's opinion, such adaptations should also take into account parameters directly related to the quality which is actually perceived by the user when such settings are used. In fact, the definition of Quality of Service (QoS) given in the ITU-T recommendation ITU-E.800 [1] clearly defines it as *"the collective effect of service performance, which determines the degree of satisfaction of a user of a service. It is characterized by a combination of service performance factors such as operability, accessibility, retainability and integrity."*

The adaptation approaches in the literature present simple adaptation algorithms which offer suboptimal solutions to the problem of selecting the application settings which better fulfill the user's expectations. Given a concrete network condition, there are many different settings that the application can change to consume less than the available bandwidth (e.g., reducing video sizes, changing video or audio codecs, reducing the quantization factor in the video codec, etc.). These adaptation algorithms just pick one from these combinations of settings satisfying the bandwidth restriction.

However, from the user's point of view, all of these possible combinations of settings do not offer the same quality. We will show that our proposal of real-time adaptive applications challenged with the notion of user-perceived QoS can lead to better adaptation algorithms being able to maintain a good level of satisfaction even when the network conditions are constantly and unpredictably changing.

Our proposed adaptation algorithm combines a genetic algorithm to decide under which network conditions the settings need to be changed and a model of the user-perceived QoS allowing the applications to select the new combination of settings which maximizes the user's satisfaction for that particular network conditions. Modeling the user-perceived QoS requires the characterization of the subjective components which define a user perception of QoS and its relation with the different parameters which define the behavior of the real-time multimedia application (i.e., codecs, rates, etc.). Given the difficulty to model the user-perceived QoS analytically, we have used the SLIPPER rule induction algorithm to generate a set of rules which models the user-perceived QoS. These rules has been extracted applying the algorithm to a large number of learning examples which have been evaluated and scored by real users. This user-perceived QoS model is then used by the application to choose which specific settings to use whenever it is told to self-adapt.

In the author's opinion, one of the most significant intended contribution of this research is the possibility to model the subjective user-perceived QoS for real-time multimedia sessions using a rule induction algorithm over an user-created set of

Manuscript received April 8, 2003; revised September 8, 2003. This work was supported by Research Projects FIT 00700002003-662 and TIC-2002-04021-C02-01. This paper was recommended by Associate Editor H. Takagi.

P. M. Ruiz is with Agora Systems S.A., E-28040, Madrid, Spain (e-mail: pedro.ruiz@agoratechnologies.com).

J. A. Botía and A. Gómez-Skarmeta are with DIIC, University of Murcia, Murcia, Spain (e-mail: juanbot@um.es; skarmeta@dif.um.es).

Digital Object Identifier 10.1109/TSMCB.2004.825912

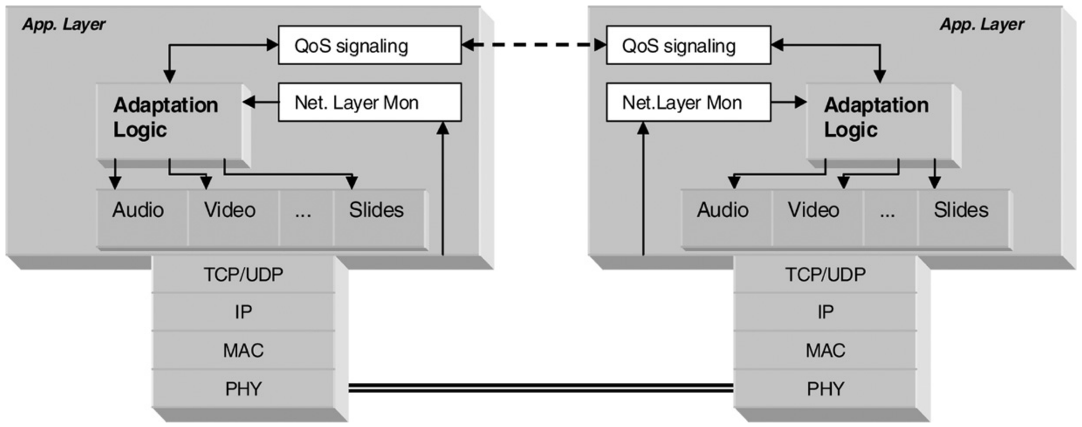


Fig. 1. Overall adaptation architecture.

scores. This is particularly interesting for adaptive applications and services, which can make use of this model to offer a higher perceptual quality to the user. Consequently, as we will demonstrate in our experiments, the user will experience a better quality when using this kind of user-aware adaptive applications than using the traditional ones.

The reminder of the paper is organized as follows. Section II presents some related work. Section III describes the architecture for adaptive applications. In Section IV, we present the optimization through genetic algorithms of the adaptation triggering. Section V explains the modeling of the user perception using the rule induction algorithm. Finally, Section VI shows some empirical results derived from the use of our adaptive application approach over an *ad hoc* network, which is one of the most challenging wireless scenarios regarding the variability of the network conditions.

II. Related Work

The idea of adaptive applications has already been used in many papers ([2], [3]) to improve end-to-end QoS in fixed networks. However, these results cannot be directly extrapolated to wireless networks, in which packet losses are not only due to congestion. In wireless links, there are many other factors which are not under control and influence the instantaneous link layer capacity.

For different kinds of wireless networks there are also interesting papers in the literature ([4]–[8]) demonstrating that adaptive applications are able to perform better than traditional multimedia applications even when the network conditions are bad. These papers use adaptation mechanisms allowing the applications to self-adapt their internal settings (e.g., changing coding schemes, frame rates, video sizes, etc.) to reduce their data rates to those that the network can support in that precise moment. However, the relation between user-perceived QoS and the data-rate required to achieve that QoS is not linear (e.g., an user could prefer a bigger video encoded using less resolution than an smaller one using high resolution even when the latter requires more bandwidth). Hence, the predefined actions used in these approaches are not able to get the better user-perceived QoS. Maximizing the user-perceived QoS requires modeling the user QoS perception. In our case, we apply machine learning to produce a rule set representing the user.

The use of machine learning approaches to solve specific issues in wireless networks has demonstrated to be effective. For example, Steinbach [9] and Brown [10] optimize power consumption of wireless interface cards by reinforcement learning [11]. Tong [12] uses reinforcement learning to minimize the probability of congestion and the probability of a call blocking. Banerjee [14] applies genetic algorithms [15] to perform QoS routing (i.e., finding paths in a network satisfying some bandwidth and delay constraints). A similar problem is solved by White [16] using the ants colony metaphor. Tang [17] applies a genetic algorithm to decide, in a video on demand application, both location and replication level of multimedia files into an array of disks. This is a multiobjective problem in which both storage consumption and blocking probability are minimized. Finally, Zhang [18] uses data mining [19] to improve security on *ad hoc* wireless networks. Using RIPPER [20], a decision rules induction algorithm, it obtains a classification model to detect fraudulent routing updates. In our case, we will apply a combination of a genetic algorithm with a more recent rule induction algorithm called SLIPPER [21] to the specific problem of maximizing user-perceived QoS in real-time adaptive multimedia applications.

III. Adaptive Application Architecture

Traditional multimedia applications are a combination of one or more components (e.g., audio, video, etc.) which use the RTP [22] protocol to transport multimedia data. These applications can use the RTCP [22] protocol to receive some feedback from the applications at the other end, but their adaptation capabilities are very limited and usually they cannot adapt at all. In order to support real adaptive applications, we add some components to the architecture of a traditional real-time multimedia applications. These new components are in charge of the signaling of QoS information, and the provision of the intelligence to keep the user-perceived QoS at an acceptable level. So, the overall architecture, as shown in Fig. 1, consists of the following components:

- multimedia application components;
- QoS signaling mechanism;
- adaptation logic.

The QoS signaling mechanism is the protocol in charge of sending and receiving reports describing the network conditions from the other end. When such a report is received, it is passed to the adaptation logic so that it can decide which internal settings the application has to use to adapt to the current network conditions while providing a good user-perceived QoS.

A. *Qos Signaling Mechanism*

The QoS signaling is a key point of the adaptation architecture as it is the only feedback that the source has from the other end. It is basically an end-to-end transport mechanism for signaling data; no special protocol is needed. In fact, it may be enough with a TCP/UDP socket between sides, or even a standard protocol like Session Initiation Protocol (SIP [23]). The QoS signaling module calculates the loss-rate and mean delay experienced by the data packets in the network. This information is carried in a special signaling packet called "QoS Report", which is sent back to the source. A sequence number is used to deal with delayed QoS Reports. The rate at which QoS Reports are sent has to represent a good tradeoff between highly dynamic adaptation and a reduced traffic load. At the moment in our implementation it is fixed at one QoS report per second. However, the use of adaptive rates is being investigated. A QoS report message presents the structure shown in Fig. 2.

Seq	% lost	Delay	User preferences	Estimated BW

Fig. 2. Components of a QoS report message.

An additional issue is that the QoS report packet itself has to traverse the network back to the server, and the probability of it actually making it there on time is inversely proportional to its importance. That is, a feedback packet is most important when it carries information about a congested network and it is not important when it is just saying that all is going well. Some experiments performed in [24] demonstrate that a UDP transport is much more appropriate to carry the feedback than a transport using TCP. The loss of QoS report messages may cause the applications not to have updated information on the network status. There are basically two approaches to overcome this problem: prioritizing QoS Reports or allowing the applications to detect such losses. As the former approach requires a tight coupling with the network layer, we have found easier to make the receivers send periodic reports toward the sources. In this way, whenever network problems come up, the adaptation logic at the sender can detect missing reports. This information can be used by the adaptation logic to implement some heuristics based on that information (e.g., downgrading the bandwidth consumption when a certain number of QoS Reports are lost).

B. *Adaptation Logic*

The adaptation logic can be seen as a function which uses the QoS Reports and additional local information to decide which settings need to be configured in each of the different multimedia components. The adaptation logic solves the problem of adapting multimedia flows to the characteristics of the different networks or terminals.

In general, most of the bad effects perceived by the user are due to packet losses. The most important input for the Adaptation Logic will be the end-to-end percentage of packet losses per reporting period. Problems due to a high jitter (defined as delay variation) may also reduce the perceived quality but this problem can usually be avoided with a proper buffer management up to a certain delay threshold. In our case, given the interactive nature of the videoconference service under consideration, packets arriving their destination later than this maximum delay are considered as packet losses. Our proposed adaptation algorithm, which is presented in Fig. 3, takes these objective facts into account. Hence, the parameters which drive the adaptation algorithm are the loss-rate to downgrade, 0% consecutive QoS Reports to upgrade and missing reports to downgrade.

Given a concrete network conditions, there is not an unique set of application settings (frame rates, sampling rates, audio and video codecs, video sizes, etc.) allowing the multimedia application to adapt its data rate to the available bandwidth. Traditional adaptive applications usually select one of the feasible combinations which reduce the loss-rate without taking into consideration the user-perceived QoS. In the ideal case, the application logic should select from all these feasible combinations the one which produces the best user-perceived QoS. Finding such an optimal set of settings requires the adaptation logic to use a model of the objective and subjective user-perceived QoS. This is something that we have accomplished (see Section V) applying a rule-induction algorithm to a user-created set of scores and preferences. As a result, a rule set modeling the objective and subjective user perception is produced.

C. *Adaptation Capabilities*

We have extended ISABEL-Lite (see Fig. 4), a reduced version of the ISABEL [25] application, to dynamically (and in real time) adapt its behavior to the available resources. The most important adaptation capabilities implemented are as follows.

1) *Coding schemes.* The application may handle contents encoded in several standards such as MPEG/H.261/H.263/MJPEG for video, or GSM/G.722/G.711 for audio. The application chooses among them at the user request or based on the information received from lower layers.
2) *Sampling rate.* Transmitting at a lower frame (or sampling) rate means saving bandwidth, and a fair quality is often achieved at less than 24 fps.
3) *Component size.* In scarce bandwidth environments the user will prefer seeing smaller videos than bad quality ones in which most of the frames are lost.
4) *Component use.* In very constrained bandwidth scenarios, the user may prefer using some components instead of using all of them with a poor quality.
5) *Buffering.* Intelligent and dynamically adaptable buffers alleviates the effects of jitter and delay in adverse network conditions. However, there is an upper delay bound (typically 500 ms) which is imposed by the interactive nature of the application.

IV. Triggering the Adaptation Procedure

There are two key decisions which define the adaptation strategy: when and how to do it. In this section, we will focus on

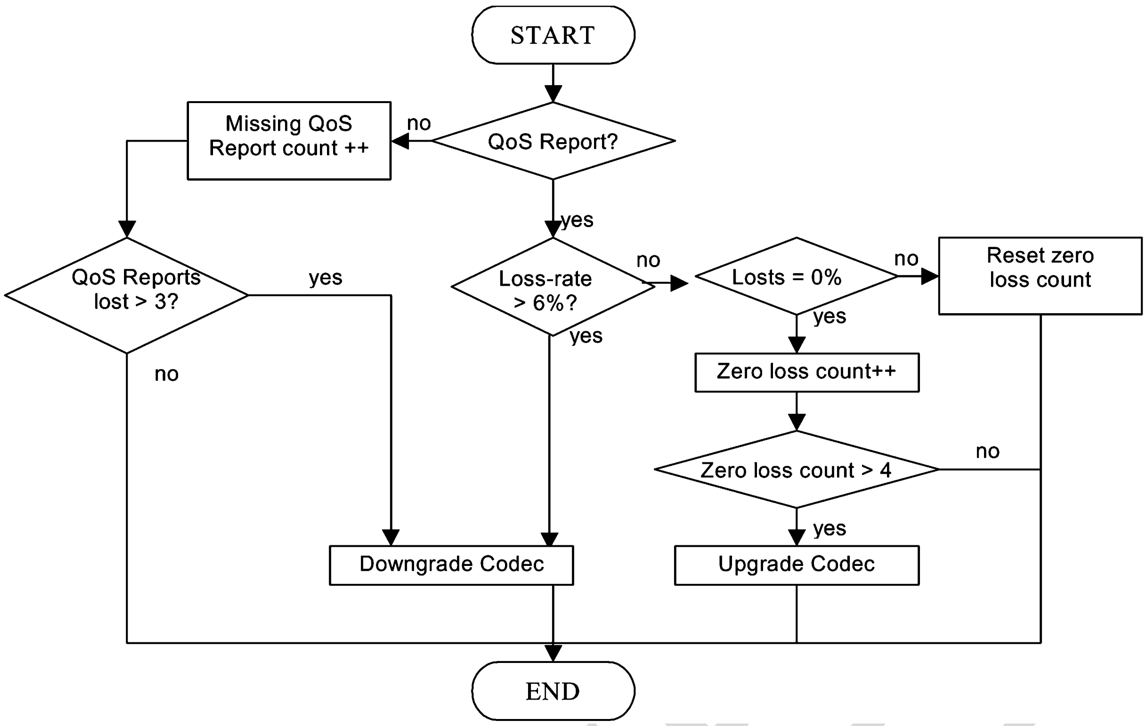


Fig. 3. Diagram of the adaptation logic.

AUDIO::PCM VIDEO::H261 (QCIF 1 16) 138
AUDIO::PCM VIDEO::MJPEG (CIF 6) 639

Fig. 4. Snapshot from our ISABEL-lite adaptive application.

the optimization of the when using a genetic algorithm whereas the how to do it strongly depends on the user perceptions of QoS, and we will study it in Section IV.

Fig. 3 illustrates the adaptation algorithm. As it can be seen, the real behavior of that adaptation highly depends on three different parameters: the loss percentage to downgrade, the number of consecutive 0% reports received before upgrading and finally the amount of QoS Reports lost which provoke a downgrading. Taking into account our empirical experiences with the application, we initially fixed the the initial values for this parameters to the values shown in the middle column of Table I.

In some other papers like [26], we have demonstrated the goodness of the adaptive applications approach as well as this concrete adaptation algorithm. As long as trying all the values for these parameters to guess the better combination would require a lot of time and it is a reduced optimization problem, we have used a genetic algorithm to calculate their optimal values. This allows us to easily fine-tune and complement the adaptation rules learned by SLIPPER.

After extensive tuning of the GA parameters, we found that a population of 30 individuals, a probability of crossover of 0.6 and 0.02 as the mutation probability were good values for our problem.

In our case, the fitness function is derived from our expertise and experiments with real users about the relative effect of the packet losses and the codec in the “user perceived QoS”. These experiments indicate that most of the bad quality perception is due to packet losses rather than the codec quality itself. So, the fitness function will behave so that the higher the packet loss-rate the lower the fitness and the higher the codec quality the higher the fitness. Hence, in (1), the fitness value is calculated as the weighted sum of two different functions: one depending on the codec quality and the other on the packet loss-rate. To gather the information required to calculate the fitness for each specific experiment, the whole RTP [22] session is logged and processed. The log contains for each packet the

TABLE I
ADAPTIVE BEHAVIORS TO BE TESTED

Parameter	1st Genotype Values	Human-Set Values	GA Values
Packet loss-rate to downgrade	35	4	2
Num. of 0%-loss reports to upgrade	4	5	10
Num. of missing reports to downgrade	3	4	3

TABLE II
AUDIO CODECS USED IN THE TRIALS

Codec	Sample Depth	Sampling Rate	Estimated BW
PCM	16 bit	8000 Hz	$\sim$ 128 Kb/s
G.711	16 bit	8000 Hz	$\sim$ 64 Kb/s
G.722	16 bit	8000 Hz	$\sim$ 40 Kb/s
GSM	16 bit	8000 Hz	$\sim$ 15 Kb/s

codec that has been used to encode the transported data. In addition, each second the loss-rate (in percentage) is stored in the log file

$$\phi_1 = w_1 \left(\sum_{i=1}^{\mathrm{PPS}*N} codec(j) \right) + w_2 \left(\mathrm{PPS} \sum_{j=1}^{N} (100 - lossrate(j))^2 \right). \quad (1)$$

Let N be the experiment duration in seconds and PPS the number of packets that the application sends per second (that in our case equals 33). Then, the total number of packets during the experiments are exactly $PPS * N$. For each packet i, the function $codec(i)$ gives discrete values between 0 and 10 000 depending on the codec used to encode the data which is transported in the packet i (e.g., for the codecs used in our experiments these values are $0 = \mathrm{GSM}$, $3300 = G.722$, $6600 = G.711$ and $10\,000 = \mathrm{PCM}$). This values has been set according to the relative perceptual quality of these codecs according to the user evaluations. So, the first part of the fitness function gives a value between 0 and $PPS * 10\,000 * N$ representing the quality of the codecs used during the experiment.

In addition, the function $lossrate(j)$ is a function which gives a value between 0 and 100 representing the packet loss rate (in percentage) for the one-second interval number j. The reason for this second term being squared is because the bad quality perception at higher packet-loss rates increases exponentially rather than linearly [2]. Finally, as this second term is calculated once a second instead of once per packet, it is multiplied by PPS. Thus, this second term of the fitness function will return values in the same range that the first term (i.e., between 0 and $\mathrm{PPS} * 10\,000 * N$). Finally, in order to give higher impact to the packet loss-rate, it is sensible to use the constants $w_1 = 0.11$ and $w_2 = 0.89$. These values have been analytically set to reflect the typical user's perceptual quality preferences [2]. That is, that an experiment using the highest quality codec all the session with high packet losses cannot get a better fitness than an experiment with a lower quality codec in which packet losses are minimal.

The results from the genetic algorithm are in Table I in the column labeled *GA values*. The column labeled as *Human-Set values* represents the manually configured parameters. Additionally, to demonstrate the improvements of the genetic algorithm we also use in our tests one of the individuals before being evolved (i.e., will have random values by default). This one is also shown in Table I in the column labeled as *1st Genotype Values.*

In order to assess the goodness of the adaptation-driven by the genetic algorithm, we have prepared a testbed scenario including different types of bandwidth changes not only in data rate but also in duration. On this testbed, we have worked with four different types of applications:

1) traditional applications which are not able to adapt to the network conditions;
2) adaptive application with human-set parameters;
3) adaptive application with nonevolved parameters;
4) adaptive application with parameters from the GA

To be sure that we are fair comparing the different kinds of adaptive behaviors, instead of performing the trials over real radio links or *ad hoc* networks, we have developed a link emulation tool which is able to reproduce specific link properties in user-configured periods of time. Additionally, we focus our trials in audio transmission which is much more sensitive to varying network conditions than other media. The trials have been performed with our extended version of the ISABEL-Lite commented before and shown in 4. The audio codecs used in our trials are summarized in Table II.

The bandwidth changes and timings which are emulated in order to assess the goodness of the different approaches are shown in Fig. 5. They have been specifically selected to be representative enough as long as they combine the different kinds of reactions which can be found in real testbeds.

The previously commented applications have been tested in this same scenario under the same emulated link conditions. Each experiment has been repeated 20 times eliminating extreme cases to be sure that the emulation is correct and no anomalous values are produced among different simulations. Fig. 6 shows the packet losses which are experienced under

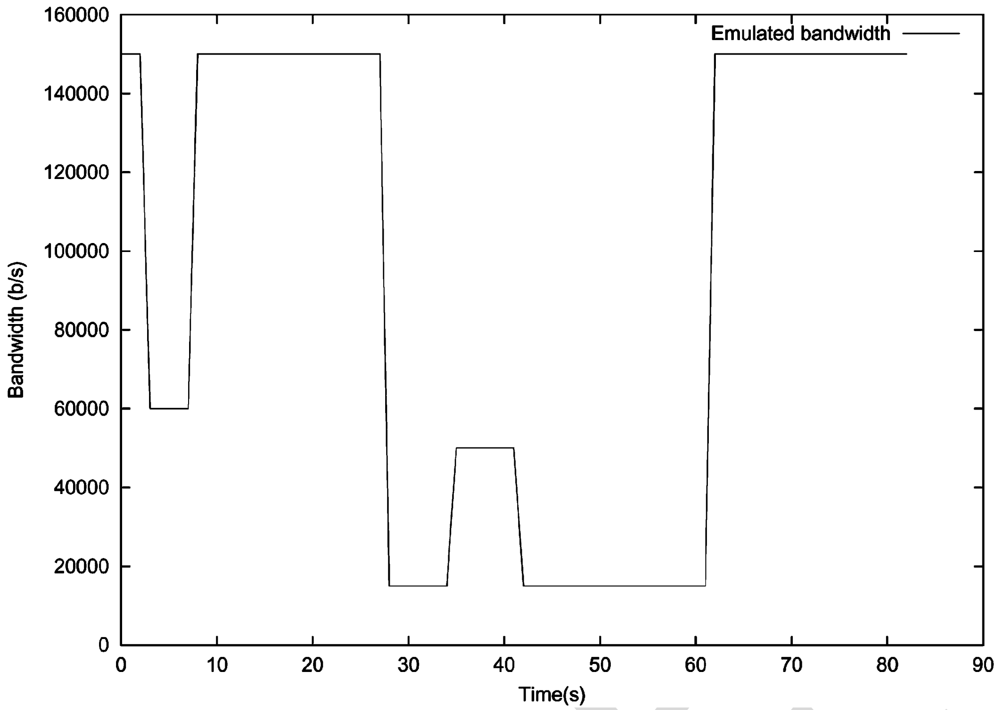


Fig. 5. Bandwidth changes emulated in the virtual link.

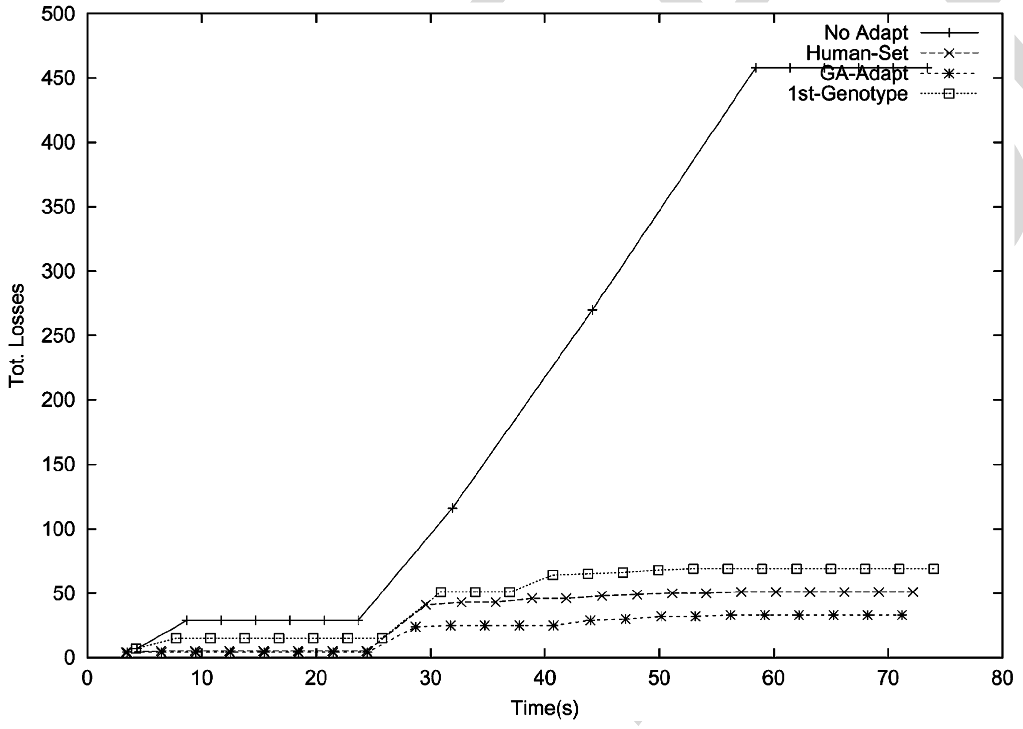


Fig. 6. Total packet losses with the different behaviors.

the three different adaptation schemes. As can be noticed, the adaptation driven by the parameters found by the genetic algorithm outperforms all the other approaches. The amount of packet losses is reduced by a factor of ten. In fact, the GA-based approach has 45% less packet losses than the human-configured adaptation.

These results are also corroborated by the loss percentages which are shown in Fig. 7. As it depicted in the figure, in low bandwidth periods most of the packets sent by the nonadaptive application are lost. However, for adaptive applications, only the first strong bandwidth reduction seems to have an important impact. In any case, this impact is reduced to less than a couple of seconds. Again, the GA-adaptation approach demonstrates that it is able to adapt better than the other approaches, with packet loss rates rarely going over 18%.

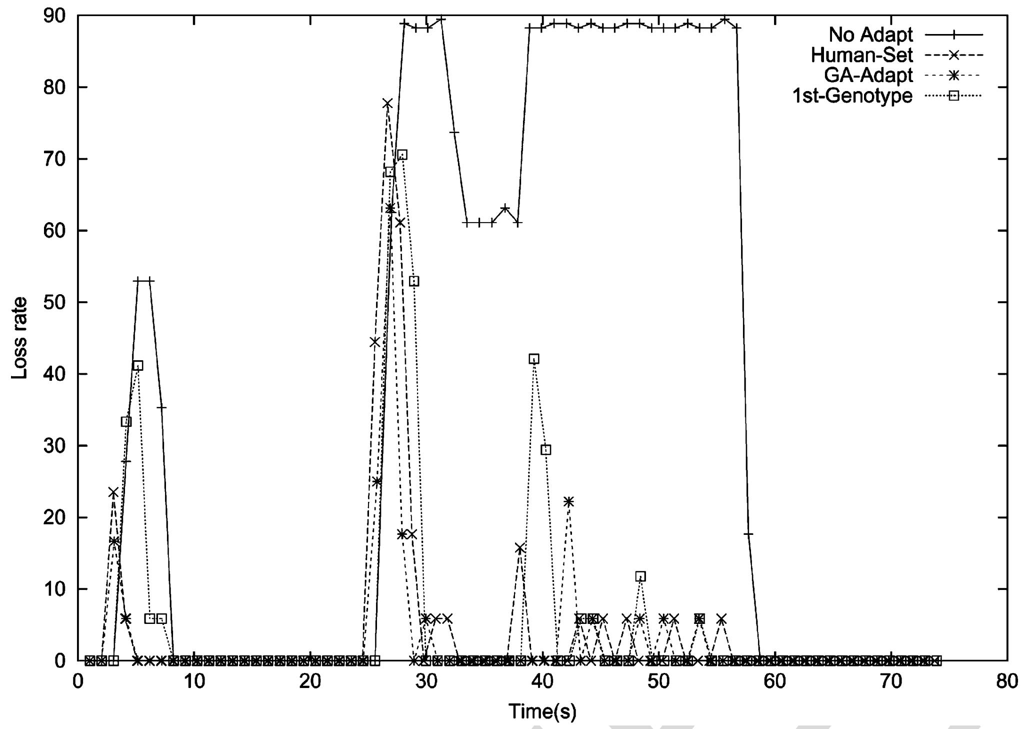


Fig. 7. Loss rate for the different approaches.

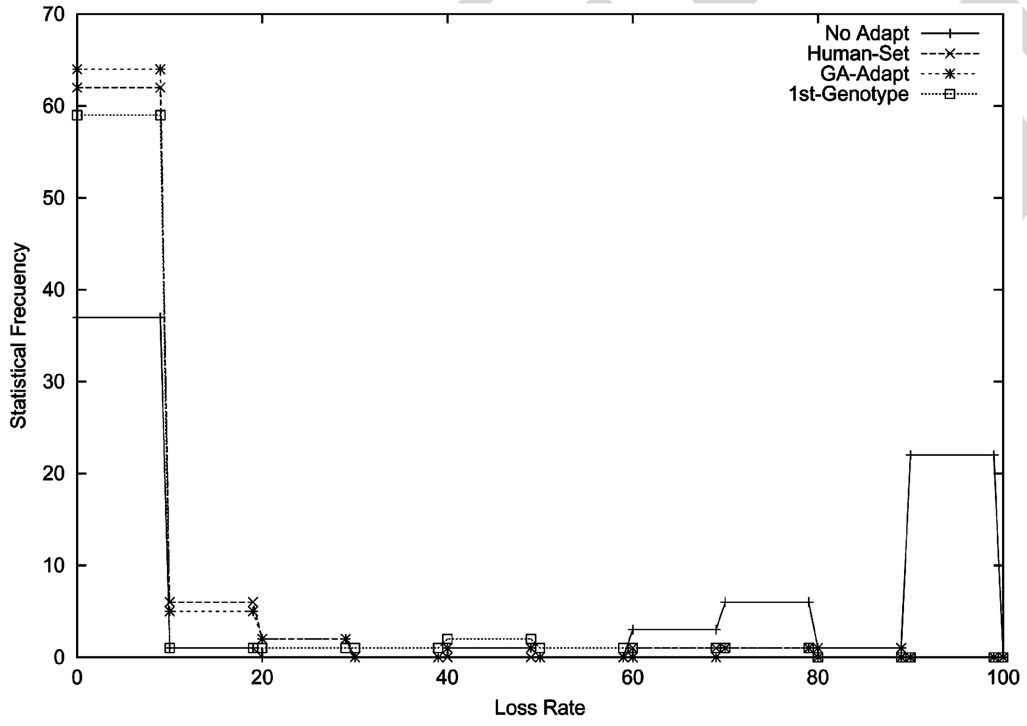


Fig. 8. Statistic histogram of loss-rate values.

The latter proposition gets clearly demonstrated in the histogram for the loss percentage distribution, which is shown in Fig. 8. This figure shows how most of the values for the adaptive case are lower than 20% of packet losses—in fact the most of the values are actually zero—while the nonadaptive application has many points going beyond 70% and most of them in the 90%–100% range.

V. Modeling QoS by Rule Induction

In Section IV, we used a genetic algorithm to decide when to trigger the QoS-upgrading and QoS-downgrading processes. In this section we focus on how to perform these upgrades so that the user-perceived QoS does not get compromised. In general, there are different combinations of settings (e.g., codecs, frame rate, video size, etc.) which allow real-time adaptive multimedia applications to adapt their bandwidth consumption to the network conditions. However, all the different combinations are not perceived with the same quality by an user. There is not a linear relation between the bandwidth and the user-perceived QoS. For example, given some specific network conditions, it might be more desirable for an user having a bigger video size rather than a higher frame rate even if both combinations consume the same bandwidth. This evaluation of user-perceived QoS depends on many subjective aspects and it is very complex to model. In this section, we present an approach to model the user perception of QoS, using a rule induction algorithm.

A. The Learning Data Set

In order to inductively model the QoS perception of an user, we have to produce a learning data set containing examples of evaluations under real networks conditions with specific multimedia application settings. Network conditions have been emulated using a multimedia reflector. This is a software tool placed in the middle of a dedicated link between two communicating nodes that will be in charge of simulating different levels of available bandwidth and packet losses. Once more we have used the ISABEL-like real-time multimedia application. This settings must be understood in terms of audio and video codecs. Audio and video codecs are in charge of capturing, coding, sending, decoding and presenting audio and video data respectively. Precisely, for the video we can also specify the size, the number of frames per second sent and a quality factor.

In the process of giving the scores to complete the learning tuples, we have tried to follow as much as possible the ITU recommendation ITU-P.800 [27] (also known as MOS recommendation) which gives recommendations on how to do the QoS evaluations, the use of values between 1 and 5 for scores, etc. A total of 864 different combinations of applications settings (in the ranges shown in Table III) have been tested, and scored by users. The table also summarizes all attributes and the corresponding range of values, which compound the data set used to model the user. The last row refers to the score given by the user.

The data set can be considered to be balanced with the following distribution of examples by score: 241 (27.8%) examples with score 1, 83 (10.4%) for score 2, 181 (20.9%) examples with score 3, 233 (26.9%) with 5 and, finally, 125 (14.46%) for the highest score.

The problem of assessing the multimedia service quality perceived by any user would be ideally solved with a function, let it be denoted with ξ, which, given a concrete user and a tuple of values for the first seven parameters of Table III, asses exactly the QoS value in $\{1, 2, \ldots, 5\}$ that user would have given. However, we will obtain an approximation $\hat{\xi}$ of ξ for a single user (i.e., the one that produced the data set mentioned above). The $\hat{\xi}$ function can be defined, as shown in (2) producing a score, given specific network conditions and specific settings for the multimedia application as follows:

$$\hat{\xi} : BW \times AudCod \times VidCod \times Size \times Qual \times FPS \times Loss \rightarrow QoS. \quad (2)$$

B. The Learning Experiments

In order to perform the learning experiments which pursue the $\hat{\xi}$ needed, we have used a distributed machine learning tool we have developed in our research group. It has been built from a software architecture called METALA (META-Learning Architecture) [28]. It is a set of software recommendations providing a methodological approach to perform typical inductive machine learning tasks.

Learning experiments have been performed using SLIPPER [21]. This algorithm does not directly use the classic search bias of divide an conquer for rule induction. Instead, it bases its strategy on boosting [29]. It uses a weak learner (i.e., a very simple rule induction algorithm) which boost by modifying learning instances probability each iteration to focus on instances not correctly classified yet. In fact, we also tested IREP, IREP* [30], and RIPPER [20]. Former algorithms which do not use boosting and all of them under-performed SLIPPER.

Two configuration parameters must be set for SLIPPER. They are the growing factor and the count of rounds to boost the weak learner we mentioned above. The growing factor is the proportion of instances from the data set used for growing a rule, meanwhile the rest is left for pruning it. We used a 80% of data for growing and the remaining 20% for pruning. Concerning the value of the second parameter, let it be denoted with t, the higher its value (from 1 to n), the higher the model accuracy and its complexity. We developed experiments with values in $\{1, 2, \ldots, 15\}$.

Results for all experiments appear represented in Figs. 9 and 10 for training and testing errors, respectively. At the moment, attention must be paid only to the curves labeled with `Orig. data`. This corresponds with training and evaluation errors on original data respectively. In these curves we have to select a concrete point with an acceptable trade off between classification error and model complexity. The experiment with $t = 5$ represents an example of such a trade-off. It has 12 decision rules. It must be noticed that a classification error estimation of 42% is a high value, indeed. However, it must be taken into account that it is not the same to classify a very good combination of network conditions and multimedia application (i.e., with a real quality of 5 for the user) as good (i.e., with a 4) than as very bad (i.e., with a 1).

We can see this fact more clearly having a look at the confusion matrix for the mentioned model. It is the following:

	1	2	3	4	5
1	**233**	3	5	0	0
2	42	**21**	19	1	0
3	21	6	**89**	58	7
4	14	2	53	**124**	40
5	7	0	10	48	**60**

where $cm[i][j] = n$ means that n examples, labeled as i where classified as j. In our particular problem, it is convenient not only that, as much as possible, all non zero values were located at the matrix diagonal but also that non zero values not located at the diagonal were near to it. For example, the cell

TABLE III
PARAMETERS CHANGED TO GENERATE USER'S SCORES

Parameter	Values	Explanation
BW	33, 56, 88, 128, 384	Limit of network bandwidth
LOSS	0..100	% loss packets
AUDCOD	PCM, G711-u, G722, GSM	Audio codec
VIDCOD	MJPEG, H.263	Video codec
FSIZE	CIF, QCIF, 160x128	Size of video frames
QFVIDEO	5, 10, 15, 30, 60	Quantization factor of video codec
FPS	2, 6, 12	Frames by second sent
QoS	1, 2, 3, 4, 5	User perceived quality

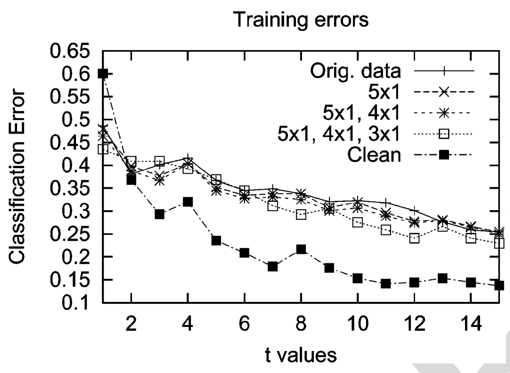


Fig. 9. Evolution of training errors for SLIPPER experiments.

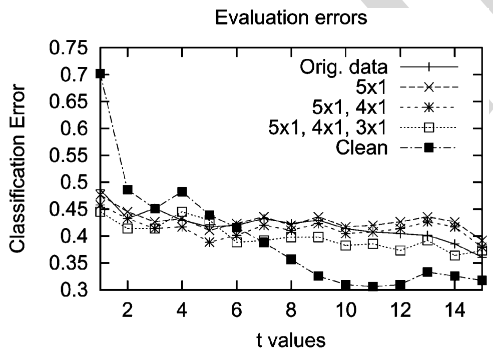


Fig. 10. Evolution of evaluation errors for SLIPPER experiments.

value of $cm[5][1] = 7$ means that seven examples, labeled with the highest quality, have been classified by the model as cases with the lowest quality. This seven particular cases have influenced the misclassification error in the same way than the seven values of $cm[3],[5]$ which refer to cases with acceptable quality and classified as very good. We have used this confusion matrix to directly select noisy examples, to correct them and after that to obtain again a new confusion matrix. Let n be the number of different classes, N the total number of examples appearing at the matrix and and $cm_{i,j}$ the value of the confusion matrix for the ith row and jth column. Then, the *condensation*, ϕ, for cm is defined by

$$\phi = \sum_{i=0}^{n}\sum_{j=0}^{n} \frac{cm_{i,j}(i-j)^2}{N}$$

and reflects the concentration of values around its diagonal. This measure will be used in order to characterize rules models behavior tuning as the data set is getting increasingly clean.

Results are shown again in Fig. 9 for training and Fig. 10 for the evaluation errors. They appear at curves labeled with 5×1 in both figures. More curves appear, labeled with $5 \times 1, 4 \times 1$ and $5 \times 1, 4 \times 1, 3 \times 1$ and represent successive corrections to noisy data with examples classified as 1 when the real score was 4 and 3, respectively. Decision rules for the final data set are represented at the same figures with label `Clean`. Notice the considerable improvement in classification errors as the number of noisy examples decrease. If we consider complexity, the number of rules also (slightly) decreases when data is more and more clean. This process is done iteratively because it implies no effort as METALA does all the work. The evolution on number of rules for models is represented at Fig. 11. We can also check that our ϕ measure for confusion matrix condensation behaves as expected if we represent it like in Fig. 12. Finally, the model we have selected from among the whole bunch of possible data sets is the one presented below. It corresponds to the cleaned data set and $t = 5$. It has a total of 12 rules. An instance is classified with a class if the sum of confidence value of all rules of the class matching the instance is higher than the negative value.

```
if matchConfidence {
[QFVIDEO >= 60, VIDCOD = MJPEG,
FSIZE = QCIF, LOSS <= 10, FPS >= 6] → 2.8792
[AUDCOD = GSM, BW >= 80,
```

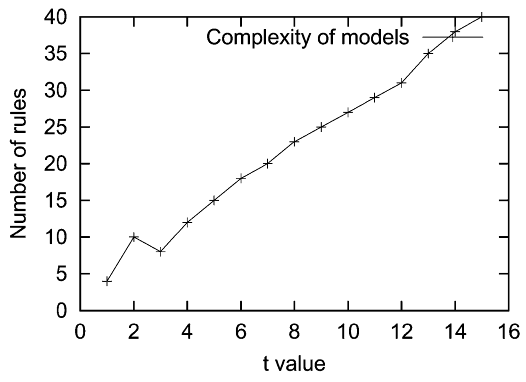


Fig. 11. Number of rules in models obtained from SLIPPER.

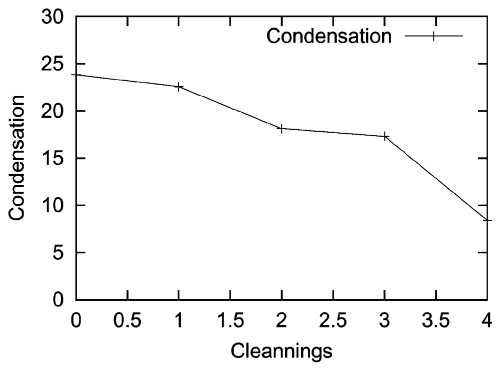


Fig. 12. Evolution in the condensation level for successive refinements on the data set.

```
QFVIDEO >= 30, FSIZE = QCIF, FPS <= 6] → 1.4357
[AUDCOD = GSM, BW >= 128, LOSS = 0,
QFVIDEO >= 30, FPS >= 3, VIDCOD = MJPEG] → 1.7013
[] → −2.4188
}> 0 then 5 else if matchConfidence{
[BW >= 384, QFVIDEO >= 40, FSIZE <= 2] → 2.7121
[QFVIDEO >= 30, VIDCOD = MJPEG,
LOSS <= 3, AUDCOD = G722, BW >= 80] → 1.1758
[FSIZE = CIF, QFVIDEO >= 30,
LOSS <= 3, AUDCOD = G722, BW >= 80] → 1.4437
[] → −1.5044
}> 0 then 4 else if matchConfidence{
[LOSS >= 30] → 2.1188
QFVIDEO <= 5] → 1.4142
[LOSS >= 16, FPS <= 3] → 1.5438
[]→ −1.098 420 727 582 606 6
}> 0 then 1 else if matchConfidence{
[LOSS >= 16] → 1.9109
QFVIDEO <= 10, FSIZE = QCIF] → 1.5861
FSIZE = 160 × 128, QFVIDEO <= 40, VIDCOD = H.263]
→ 1.5861
[] → −0.3953
}> 0 then 2 else 3
```

The rule set extracted by SLIPPER allows us to identify which individual settings are most important for the user-perception of QoS and which is the relation between them. For example, the rules imply that the higher the frame rate the better the quality, but the user prefers changing from a higher frame rate to a lower one, provided that the video size is increased. Although there are big differences in bandwidth consumption between the different audio codecs, the rule set has identified that provided that there are no losses in the network, the better option is to use GSM only when the network resources are scarce, and using G.722 is enough in our low-bandwidth scenario to guarantee a good user-perception. This information given by the rule set, has allowed us to generate a concrete combination of settings, among which, the application will change depending on the network conditions. The derivation of these settings from the rules, guarantees a good user-perceived QoS. This combination of settings is shown in Table IV.

TABLE IV
QUALITY STEPS FOR THE REAL-TIME ADAPTIVE APPLICATION

Step	A. Codec	V. Codec	V. Rate	V. Size	Q	Est. BW
0	GSM	-	0 fps	-	-	20 Kbps
1	GSM	MJPEG	4 fps	160x128	50	80 Kbps
2	G.722	MJPEG	8 fps	160x128	40	140 Kbps
3	G.722	H.263	6 fps	QCIF	50	190 Kbps
4	G.722	MJPEG	4 fps	CIF	30	230 Kbps
5	G.722	MJPEG	6 fps	CIF	50	350 Kbps

VI. EMPIRICAL RESULTS

In order to evaluate the effectiveness of our proposal, we have set up a real wireless *ad hoc* testbed, on which we will compare the performance of real-time videoconferencing both with traditional applications and with adaptive applications. The testbed has been deployed in the basement of the CS Faculty at the Univ. of Murcia (see Fig. 13). We use the Multicast MANET Ad hoc Routing Protocol (MMARP [31]), which is a new Multicast *ad hoc* routing protocol that we proposed. It supports both *ad hoc* nodes as well as standard IP Multicast nodes. MMARP nodes are numbered from 1 to 4, R is a standard-IP multicast receiver and S is a standard-IP multicast source. The source (S) follows (at walking speed) the path which is shown in Fig. 13 while it runs a videoconference session with node (R).

The route has been specifically selected so that link breaks and MMARP route changes take place during the videoconferencing session. Furthermore, the signal strength changes due to the variation of the distance to MMARP nodes and the number of intermediate walls to traverse. This makes the available bandwidth vary during the session.

The trials have been performed using our MMARP implementation for Linux. It is a user-space daemon which handles MMARP packets before they are processed by the TCP/IP stack. In addition, we have also extended the RTP-based ISABEL-lite

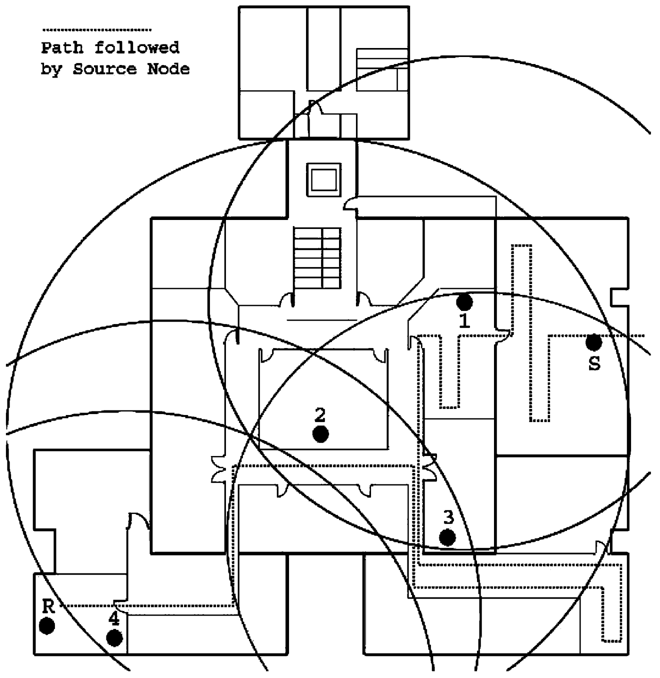


Fig. 13. Map of the testbed scenario.

videoconferencing application to use our adaptive application framework. The settings which are used by the application as adaptation steps are those extracted from the rule set which models the user-perception. This guarantees that the user-perceived QoS will be maintained as a consequence of keeping the loss-rate at a minimum. These settings are presented in Table IV. In order to guarantee a fair comparison of both approaches, the adaptive application starts with the quality step number 3, which is the only one used by the nonadaptive application, and is around the mean bandwidth which we calculated during the whole session. The results which we present are extracted from the RTP traces which are generated by the videoconferencing application. We have used the same route, at the same speed and in the same network conditions for the adaptive and non-adaptive trials.

The results presented in Fig. 14 show that the use of adaptive applications is able to reduce the overall packet losses both for audio and video to approximately one third. As expected, the differences are higher in the periods in which there is less bandwidth available. This is also noticed in the variation of the delays depicted in Fig. 15. In the same critical periods, the nonadaptive approach is not able to control the growing of the end-to-end delay, whereas the adaptive one is able to quickly restore the original state.

The overall packet losses is a good reference to identify the points of the trial in which the network conditions are most critical. This is identified by an increase in the slope of the total packet loss curve. However, what really affects the user perception of QoS is the instantaneous loss-rate, which is what causes the service disruptions. For example, Bolot [2] identifies an instantaneous 20% of packet losses as the point in which an audio flow can be considered of poor quality.

In Fig. 16, we compare the statistical histogram for the distribution of the audio loss-rate for both approaches. The same statistical analysis is performed for the video flow in Fig. 17. For example, for the audio flow, the adaptive application approach is able to keep the loss-rate below 10% all the time. In fact, it keeps the loss-rate below 5% during the 91% of the time. For the video flow, the loss-rate is kept under the 5% the 64% of the time, and its has been under the 10% the 78% of the time.

These figures clearly demonstrate that the adaptive application approach on top of our MMARP implementation has been able to offer a very good user-perceived QoS in a scenario in which traditional multimedia applications offer a highly variable quality. Furthermore, both for audio and video flows the adaptive applications approach highly reduces the loss-rate, demonstrating that adaptive applications are a good approach for dealing with the changing network conditions which characterize *ad hoc* networks.

Although both Figs. 16 and 17 demonstrate that the user-perceived QoS is improved when using our novel machine learning driven adaptation, and that is also supported by the validity of the rules modeling the user's QoS perception, we have conducted an additional test with real users giving scores over these network scenarios. As in the generation of the learning examples (see Section IV), we have tried to follow the MOS ITU-recommendation [27] as much a possible. Eight different users have given their QoS evaluations with scores in between 1 and 5. The higher the score the better the quality (i.e., 1 = poor, 2 = bad, 3 = fair, 4= good, 5 = excellent). Fig. 18 shows the mean evaluation from these 8 users in different points of the path and finally an overall QoS perception value. These points are identified by the elapsed time since the starting. These specific instants have been specifically selected in points in which the network conditions are most critical.

As it is depicted in the figure, the machine-learning driven adaptation logic clearly outperforms the traditional multimedia adaptation because the proposed approach is able to maintain a good QoS level even when the network resources are extremely scarce and variable. As can be observed in Fig. 18, the mean overall perception for our adaptive approach is around 3.87 which means that most of the users scored the overall QoS as good. However, the nonadaptive approach has scored in average 1.87 which means that most of the users scored the overall QoS as bad. In general, even in the individual evaluations, the machine learning driven adaptation has rarely been scored on average below the good QoS perception, and never below the fair QoS perception. These results are extremely good taking into account that mobile *ad hoc* networks are one of the most extreme cases of limited bandwidth, link variability and error-prone links. They fully support the conclusion that machine learning driven adaptive multimedia applications offer an excellent performance compared to traditional approaches.

VII. Conclusions and Future Work

We have proposed a novel machine learning driven adaptation approach for real-time adaptive multimedia applications. This approach, which is based on a combination of a genetic algorithm and the SLIPPER rule induction algorithm, has demonstrated to outperform both traditional real-time multimedia applications and manually-tuned adaptive multimedia

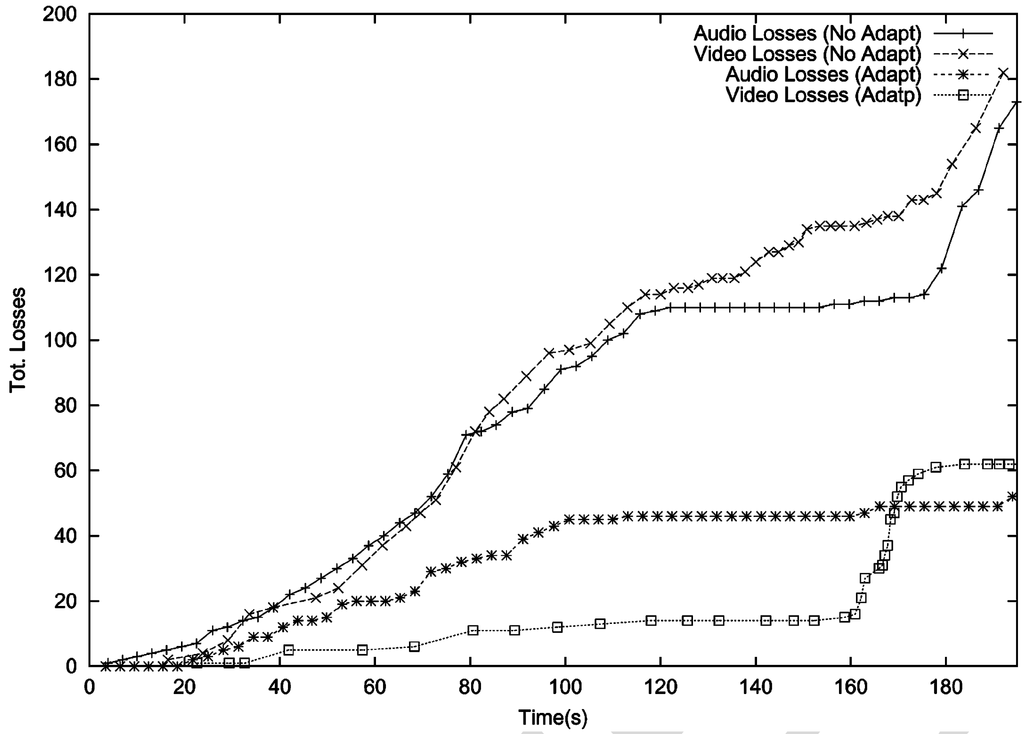


Fig. 14. Variation of the total losses over time.

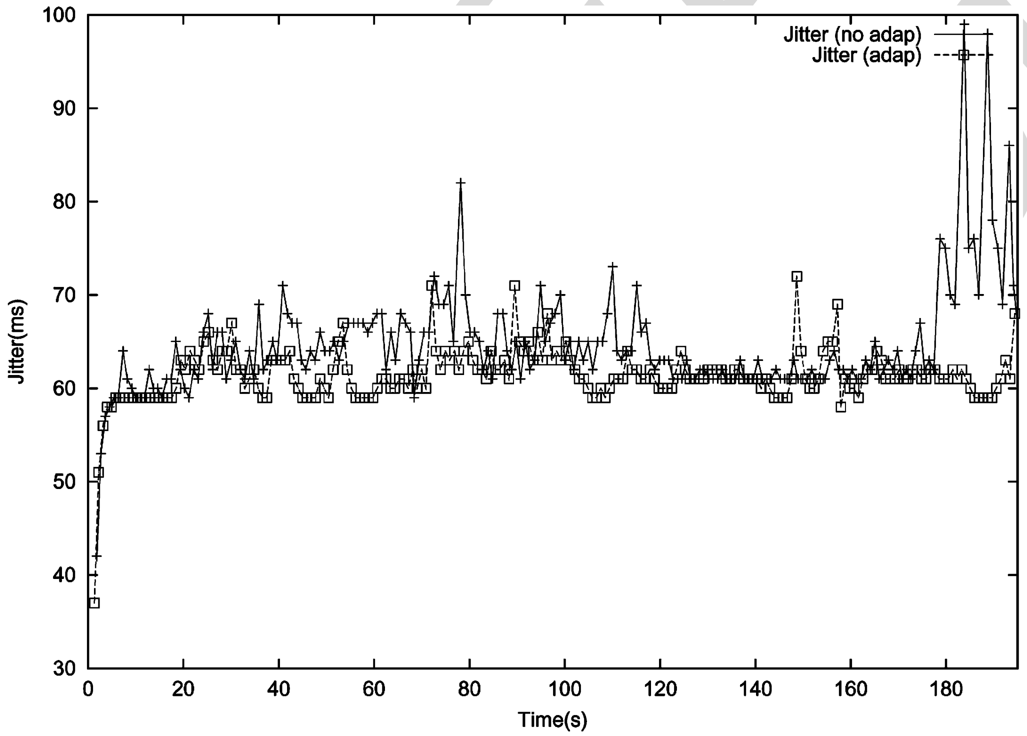


Fig. 15. Inter-arrival jitter over time.

applications. Both the genetic algorithm and the rule induction algorithms have demonstrated to be effective when used both separately and together. However, questions regarding the relation between the QoS perceived by an user and the space compound by the configuration parameters of a multimedia application remain open. For example, what is the shape of this multidimensional space? This is an important question in machine learning. If this question is correctly answered, the inductive bias [13] may be choosen to obtain a model which best fits that space.

We have used a mobile *ad hoc* network testbed, to quantify the goodness of this approach in mobile and wireless scenarios in which the QoS cannot be guaranteed at the network layer. In addition to the adaptation of the data rates generated by the appli-

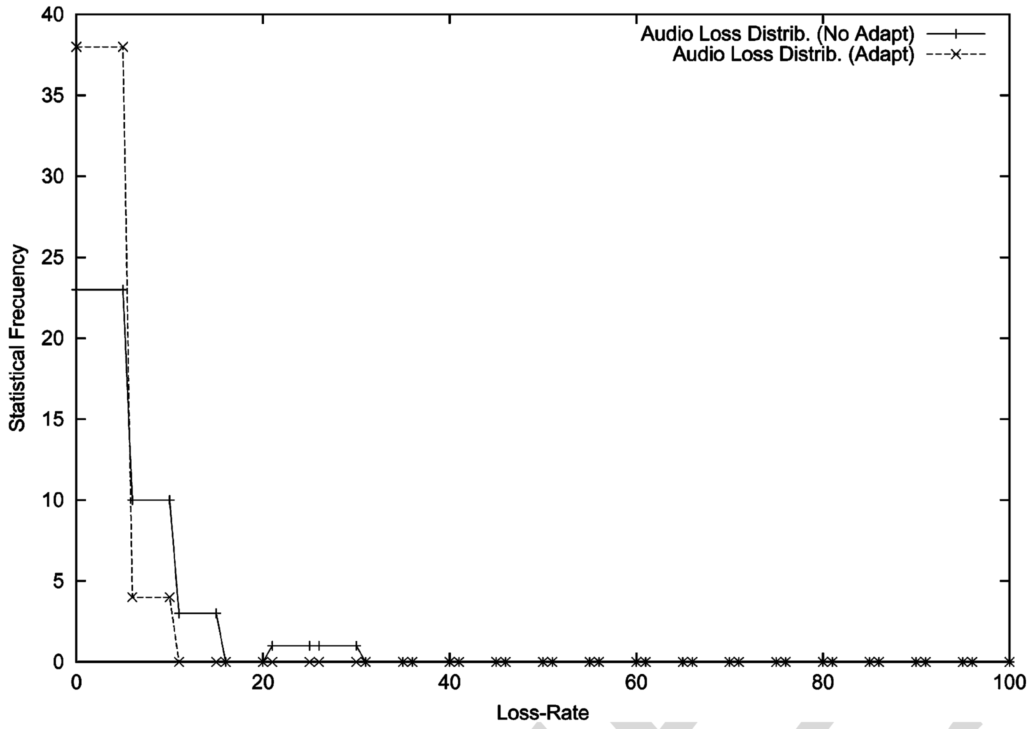


Fig. 16. Histogram for audio loss-rate.

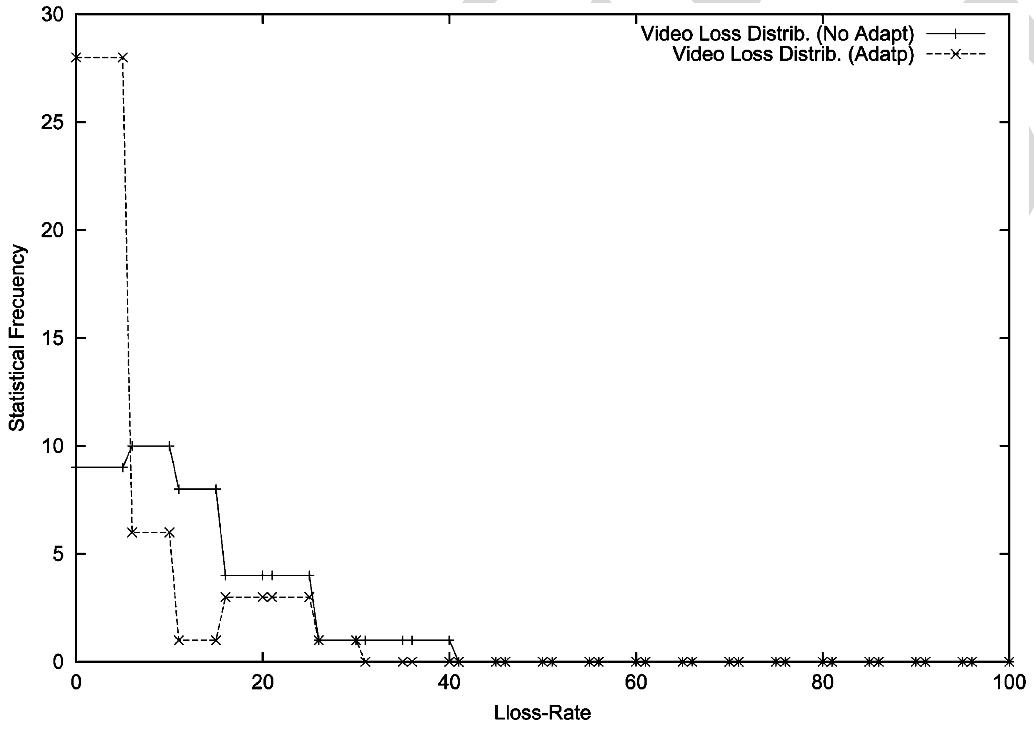


Fig. 17. Histogram for video loss-rate.

cations to the network conditions, our machine learning driven adaptation approach has been able to offer a good user-perceived QoS even in situations in which traditional applications perform badly. These results are clearly supported by the evaluations which the users have done on the overall performance in the wireless *ad hoc* testbed.

As a future work, we are extending this adaptation to the network conditions toward a fully ambient intelligence platform [32]. The idea would be to support real-time adaptation not only to the network conditions, but also to the user's context including preferences, terminal resources, terminal capabilities, and user's location, among other context information. We are using the multiagent programming paradigm [33] for that. Cur-

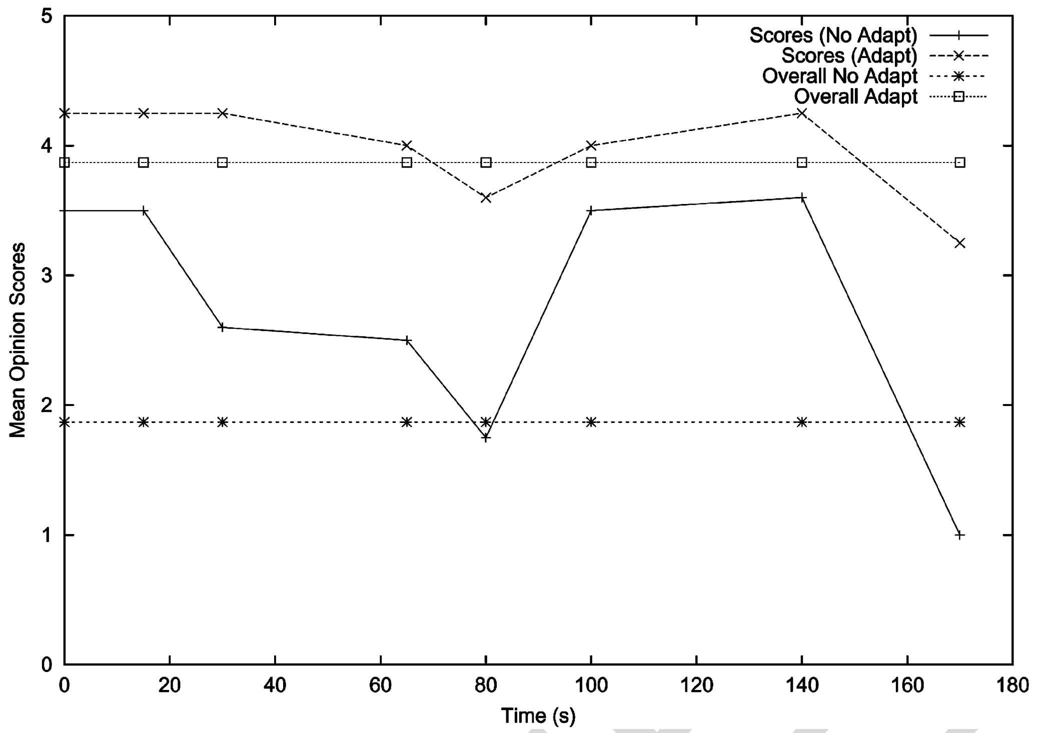


Fig. 18. User scores to during the path in the *ad hoc* testbed.

rently, we are definning the ontology which represents users, multimedia applications capabilities, QoS, and network monitoring parameters. Implementation is being done by using a standard agent programming platform, JADE, running on a Java virtual machine.

References

[1] *Terms and Definitions Related to Quality of Service and Network Performance Including Dependability*, ITU-T Recomendation E.800 (0894), **AUTHOR: PLS. SUPPLY YEAR**.

[2] J.-C. Bolot and A. Vega-Garcia, *The Case for FEC-Based Error Control for Packet Audio in the Internet*: ACM Multimedia Systems, 1998.

[3] D. Sisalem, “End-to-end quality of service control using adaptive applications,” in *IFIP Int. Workshop on Quality of Service*, **AUTHOR: PLS. SUPPLY LOCATION**, 1997.

[4] M. Kazantzidis, S.-J. Lee, and M. Gerla, “Permisible throughput network feedback in AODV MANETSs,” in *Proc. ICC 2001*, Helsinki, Finland, June 2001.

[5] T.-W. Chen, M. Gerla, M. Kazantzidis, Y. Romanenko, and I. Slain, “Experiments on QoS adaptation for improving enduser speech perception over multihop wireless networks,” in *Proc. QoS Mini Conf. in Conjuntion With IEEE ICC’99*, Vancouver, BC, Canada, June 1999.

[6] A. Alwan, R. Bagrodia, N. Bambos, M. Gerla, L. Kleinrock, J. Short, and J. Villasenor, “Adaptive mobile multimedia networks,” *IEEE Personal Commun.*, pp. 34–51, Apr. 1996.

[7] M. Mirhakkak, N. Schult, and D. Thomsom, “Dynamic bandwidth management and adaptive applications for a variable bandwidth wireless environment,” *IEEE J. Select. Areas Commun.*, vol. 19, pp. 1985–1997, Oct. 2001.

[8] R. Ramanathan and R. Hain, “An ad hoc wireless testbed for scalable, adaptive QoS support,” *IEEE WCNC*, pp. 998–1002, Nov. 2000.

[9] C. Steinbach, “Adaptive mid-level power management for wireless devices,” in *Proc. SOW Conf.*, **AUTHOR: PLS. SUPPLY LOCATION**, July 2002.

[10] T.-X. Brown, “Low power wireless communication via reinforcement learning,” in *Advances in Neural Information Processing Systems*, S. A. Solla, T. K. Leen, and K.-R. Muller, Eds. Cambridge, MA: MIT Press, 2000, pp. 893–899.

[11] R. Sutton and A. Barto, *Reinforcement Learning*. Cambridge, MA: MIT Press, 1998.

[12] H. Tong and T.-X. Brown, “Reinforcement learning for call admission control and routing under quality of service constraints in multimedia networks,” *Mach. Learn.*, vol. 49, no. 2-3, Dec. 2002.

[13] J. F. Provost and B. G. Buchanan, “Inductive policy: The pragmatics of bias selection,” *Mach. Learn. J.*, vol. 20, pp. 35–61, 1995.

[14] N.-N. Banerjee and S.-K. Das, “Fast determination of QoS-based multicast routes in wireless networks using genetic algorithm,” in *Proc. IEEE Int. Conf. Communications*, vol. 8, 2001, pp. 2588–2592.

[15] J.-H. Holland, *Adaptation in Natural and Artificial Systems*. Cambridge, MA: MIT Press, 1975.

[16] T. White, B. Pagurek, and F. Oppacher *et al.*, “ASGA: Improving the ant system by integration with genetic algorithms,” in *Proc. 3rd Annu. Conf. Genetic Programming 1998*, J. R. Koza *et al.*, Eds., Madison, WI, 1998, pp. 610–617.

[17] K.-S. Tang, K.-T. Ko, and E.-W.-M. Wong, “Optimal file placement in VOD system using genetic algorithm,” *IEEE Trans. Ind. Electron.*, vol. 48, pp. 891–897, 2001.

[18] Y. Zhang and W. Lee, “Intrusion detection in wireless ad-hoc networks,” in *Proc. 6th Annu. Int. Conf. Mobile Computing and Networking (MobiCom’2000)*, New York, Aug. 2000, pp. 275–283.

[19] M. Holsheimer and A.-P.-J.-M. Siebes, “Data mining: The search for knowledge in databases,” Comput. Sci./Dept. Algorithmics and Architecture, **AUTHOR: PLS. PROVIDE SCHOOL NAME, CITY, AND COUNTRY**, Tech. Rep. CS-R9406 1994, 1994.

[20] W.-W. Cohen, “Fast effective rule induction,” in *Proc. 12th Int. Conf. Machine Learning*, 1995, pp. 115–123.

[21] W.-W. Cohen and Y. Singer, “A simple, fast, and effective rule learner,” in *Proc. Conf. American Asociation for Artificial Ingelligence*, **AUTHOR: PLS. SUPPLY LOCATION/PAGES**, 1999.

[22] H. Schulzrinne, S. Casner, R. Frederik, and V. Jacobson, “RTP: A transport protocol for real time applications,” in *IETF*, Jan. 1996, RFC 1889.

[23] M. Handley, H. Schulzrinne, E. Schooler, and J. Rosenberg, “SIP: Session Initiation Protocol,” Internet Engineering Task Force, Request for Comments RFC 2543, 1999.

[24] M. Kazantzidis, L. Wang, and M. Gerla, “On fairness and efficiency of adaptive audio application layers for multihop wireles networks,” in *Proc. IEEE MOMUC’99*, San Diego, CA, Nov. 1999.

[25] The ISABEL CSCW Application. [Online]. Available: http://www.agora-2000.com/productos/isabel.html

[26] P.-M. Ruiz and E.-J. Garcia, “Improving user-perceived QoS in mobile and wireless IP networks using real-time adaptive multimedia applications,” in *PIMRC’2002*, vol. 3, Lisbon, Portugal, Sept. 2002, pp. 1467–1471.

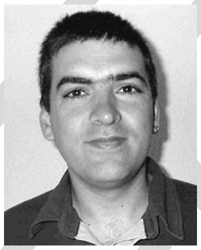
[27] *Subjective Quality Test Based on Mean Opinion Scores (MOS)*, ITU-T Recomendation P.800, 1996.
[28] J. A. Botia, A. Gomez-Skarmeta, M. Valdes, and A. Padilla, "Metala: A meta-learning architecture," in *Proc. 7th Fuzzy Days*, Dortmund, Germany, Oct. 2001.
[29] Y. Freund and R.-E. Schapire, "A short introduction to boosting," *J. Jpn. Soc. Artific. Intell.*, vol. 14, no. 5, pp. 771–780, Sept. 1999.
[30] J. Furnkranz, "Separate and Conquer Rule Learning," Austrian Res. Inst. Artificial Intelligence, Vienna, Austria, 1996.
[31] P.-M. Ruiz, A. Gomez-Skarmeta, and I. Groves, "The MMARP protocol for efficient support of standard IP multicast communications in mobile ad hoc access networks," in *Proc. 1st Mobile & Wireless Summit*, vol. II, Aveiro, Portugal, June 2003, pp. 478–482.
[32] G. Riva, P. Loreti, M. Lunghi, F. Vatalaro, and F. Davide, "Presence 2010: The emergence of ambient intelligence," in *Being There: Concepts, Effects and Measurement of User Presence in Synthetic Environments*, G. Riva, F. Davide, and W. A. Jsselsteijn, Eds. Amsterdam, The Netherlands: IOS Press, 2003.
[33] M. Wooldridge, *An Introduction to MultiAgent Systems*. New York: Wiley, 2002.

Pedro M. Ruiz in Murcia, Spain, in 1975. He received the B.Sc. and M.Sc. degrees in computer science in 1996 and 1999, respectively, and the Ph.D. degree in telematics in 2002, all from he University of Murcia.

From 1999 to 2001, he coordinated several research activities in the Spanish National Research Network (RedIRIS). He was also part-time Assistant Professor at the Telecomunicactions Engineering Department, University Carlos III, Madrid, Spain. In May 2001, he joined Agora Systems S.A., Madrid,where he is an R&D Manager. Currently, he is also teaching and coordinating research activities in the Department of Information and Communications Engineering, University of Murcia. His main research interests include advanced network services and protocols, wireless IP networks, mobile and wireless *ad hoc* networks, and distributed applications and services.

Juan A. Botía was in Murcia, Spain, in 1972. He received the B.Eng. and Ph.D. degrees in Computer Science from the University of Murcia in 1996 and 2002, respectively.

From 1996 to 1997, he was a Senior Engineer with Tecnatom, S.A. From 1997 to 1999, he was an Associate Professor with the University of Alcala, Madrid, Spain. Since 1999, he has been an Assistant Professor in the Department of Information and Communications Engineering, University of Murcia. His main research interests include machine learning and distributed AI.



Antonio Gómez-Skarmeta was born in Santiango de Chile, Chile, in 1965. He received the M.S. degree from the University of Granada, Granada, Spain, and the Ph.D. degree from the University of Murcia, Murcia, Spain, both in computer science.

He is an Associate Professor in the Department of Information and Communication Engineering (DIIC), University of Murcia. His research interests are fuzzy modeling and hybrid systems, soft-computing techniques, intelligent systems, intelligent agents, distributed services, network protocols, and security.