A QoS Metric for Packet Networks

Goncalo Quadros, Edmundo Monteiro, Fernando Boavida

Universidade de Coimbra, Departamento de Engenharia Informática
Polo II, Pinhal de Marrocos, 3030 Coimbra, PORTUGAL
{quadros, edmundo, boavida}@dei.uc.pt

ABSTRACT

Several approaches have been proposed to empower communication systems with quality of service (QoS) capabilities. In general, their main goal is to coherently support the end-to-end performance needs of applications, based on the establishment of, and agreement on, a set of concepts, policies and mechanisms. Regardless of the used approach, an important challenge associated with quality of service provision is the development of an efficient and flexible way to monitor QoS.

The existence of an effective metric to quantify the QoS offered to classes or flows of data, and to assess the performance of communication systems, would facilitate the implementation of a QoS monitor. Such a QoS metric should be able to produce comparable measures, independently of the nature and scope of the objects to quantify, that is, should turn possible uniform QoS measures. Nevertheless, the main difficulty related to the development of such metric steams exactly from the disparate nature and scope of the things to measure (typically, throughput, transit delay or loss).

This paper discusses the above mentioned difficulties and proposes a QoS metric intended to support QoS measurements on packet switching networks. In our opinion, the proposed metric will facilitate the development of an effective and truly integrated QoS management system, which is fundamental to construct QoS-capable communication systems able to efficiently deal with the increasing variety of applications.

This paper also presents the main challenges found during the first approach to the QoS metric implementation. The intention was to test the metric basic concepts, to assess its feasibility, and to measure its associated overhead. The results of these overhead tests are also presented.

Keywords: QoS Metric, QoS Measuring, QoS on Packet Switching Networks.

1. INTRODUCTION

The quality of service concept is at the basis of an intense work in the computer and communication investigation arenas. Dealing with QoS is to deal with a large variety of issues and factors related to the performance of technologies used in information systems and to their impact on users.

Interactivity with human users imposes limits on information transit delay. Isochronous or continuous traffic, produced by multimedia applications, imposes limits on *jitter*. The bursty nature of compressed video requires available bandwidth in the communication system. Conversely, common nowadays applications, for example e-mail or file transfer, have strict reliability requirements.

The heterogeneous traffic produced by all these distinct applications will tend to use the same communication system in the near future, which has to behave differently according to the characteristics of carried data. Basically, applications expect distinct performance in terms of data transport delay, bandwidth, reliability, failure recover, or session establishment latency [Partridge93].

Different approaches can be used to construct QoS-capable systems. One would be to over-dimension the system, using the requirements put by worst load conditions, supporting the resulting waste of resources. Another would be to use scale and filter mechanisms to adapt the generated load to the available resources, assuming that applications are prepared for that adaptation and for the possibility of not always having enough quality of service [Steinmetz97]. Yet another approach would be to do resource reservation, and/or to use special data scheduling techniques, for processing data from source to destination hosts. We believe, like R. Steinmetz and L. Wolf, that distributed computer systems will be for a long time in the "window of scarcity" and so, to provide adequate QoS to applications, the later approach must be used. In line of this approach, grossly two techniques are being investigated:

-techniques based on QoS specification and resource reservation, used to assign a certain part of the communication resources to each flow of data;

-techniques based on QoS differentiation, which assume that traffic is divided into classes that are to be treated differently by the communication system; regardless of the available capacity, certain classes receive higher QoS than the others.

It is our believe that the very first step to construct efficient and skilled QoS-capable systems should be the conception of an effective way to measure QoS. Such a measurement mechanism will certainly facilitate the control and management of the provided QoS, enabling the construction of better and more efficient systems, with higher functionality and lower complexity. In any way, the lack of proposals to consistently measure QoS is notorious, and the approach proposed by this paper tries to address this gap.
This paper presents a way to measure quality of service in packet switched networks, more precisely it proposes a QoS metric to support that task. The following sections discuss the usefulness of such a metric and detail our proposal for it. They present also the main difficulties found during the foremost approach in its development, which was made at IP level in a FreeBSD system running as a router. Finally, the paper presents the results of the metric overhead tests made to the prototype constructed at LCST-UC[1].

2. DO WE NEED A QOS METRIC? WHICH METRIC?

The main goal of a typical packet switched communication system able to provide quality of service is to maximize the satisfaction of its users by efficiently employing the available resources. To conveniently support applications with diverse nature and specific QoS requirements, it is unavoidable to use a comprehensive set of management and control functions, or mechanisms, such as the ones related to scheduling, policing, or admission control tasks. These mechanisms have a dynamic/adjustable behavior, and their fine tuning is paramount for the performance of QoS-capable communication systems. Also important is the ability to integrate the management of all the mechanisms related to QoS provision.
The results of management and control tasks depend on the richness of the available information on the system dynamics, at macroscopic and microscopic levels. Furthermore, the typical *best effort* paradigm, associated with the operation of nowadays communication systems, is changing to one based on performance compromises, explicit in service contracts or implicit, for instance, in the operation of the differentiated services model. That is also a very good reason to develop ways for collecting meaningful information on the communication system, which includes the QoS that the communication system is providing.
Therefore, QoS monitoring activities should supply a broad, consistent and sufficiently detailed view on the quality of service that is in fact given to applications. Nevertheless, the varying nature of the objects to measure and the strong dependence of the meaning of measures on factors such as the involved application types, turns difficult to gather extensive and meaningful information on the communication system operation. Absolute measures (which would produce values in μs for delay, *bps* for througput, a percentage for losses, etc.) are little more than useless.
The way to follow is to conceive a metric able to provide normalized and comparable QoS measures, that explicitly considers the different nature of the measures and their meaning. We are proposing such a metric. A metric whose main goal is to quantify QoS, considering its dissimilar facets, that is, the different quantifiable QoS characteristics [ISOIEC9680], as throughput, delay, loss or *jitter*, taking into account the following:
- all the QoS measures should be comparable;
- the QoS measures should reflect the supplier, or communication system, point of view and, in addition, the consumer, or application, point of view;
- it should be possible to get measures with different granularities, in order to satisfy the diverse needs of distinct measure addressees (for instance, a human being, or some mechanism related to packet or cell processing);
- it should be possible to get measures with different scopes, such as: the end-to-end QoS given to one data flow; the contribution of each communication system module[2]; the (global) performance of a module, etc.;
- the metric should be based on simplicity and scalability;
- the metric should be independent of any QoS architecture or technology.

For short, we expect that our metric can provide overall information on the communication system operation, enabling its manager to anticipate any performance problem or to accomplish its general tuning. We also expect that the metric can provide refined and consistent information on the operation of low-level communication system mechanisms, allowing their fine and dynamic regulation. In addition, we think that the proposed metric, given that it provides normalized measures, will facilitate the fully integration of all the management and control tasks related to QoS provision, and thus will be an important piece in the construction of extremely effective QoS-capable communication systems.
The metric is presented in the following sections. It was conceived to be applied to intermediate layers and to lower layers of communication systems. Nevertheless, its concepts can be easily applied to other systems where the need to measure the provided QoS does exist (for instance, operating systems, or communication system higher layers). Therefore, the metric

[1] Communications and Telematic Services Laboratory of the University of Coimbra.
[2] Later, we will introduce the concept of module; for now it can be thought of as a piece of equipment.

can be applied to comprehensive QoS architectures, providing an integrated and consistent way to measure quality of service in networks and end-systems.

3. BASIC CONCEPTS AND GENERAL ASSUMPTIONS

The proposed metric is based on the following main principle: *the quality of service supplied by a communication system should be measured according to its impact on final entities, namely QoS consumers (applications/users) or QoS providers (communication systems/service suppliers).*

This principle embodies a natural way to measure quality of service and establishes a criterion to normalize QoS measures. From the applications point of view, quality of service fundamentally means absence of congestion. Applications need, at least, a minimum QoS level and they work equally well if the QoS provided is better than the expected one. On the other hand, from the communication system point of view the goal is to conveniently support the applications without resource waste. Therefore, it is important to constantly monitor QoS , in order to continuously provide the strictly necessary QoS.
For short, a QoS metric must furnish a way to measure QoS deviations from what it is considered normal in one and the other directions (i.e., QoS degradation and QoS excess). More precisely, using the above mentioned principle, a QoS metric must provide a way to measure the impact of QoS deviations on final entities that are sensitive to QoS changes.
In order to achieve this goal, the very first challenge to overcome is to find a way to measure the impact of QoS changes on QoS consumers and suppliers. Our proposal is to use the concept of measurement zones. These determine - for each QoS characteristic and for each communication system point where we want to measure - the impact on applications/users, or on the communication system, as a function of the QoS characteristic value[3].
According to the proposed metric, the value of a given QoS characteristic, in a given communication system point, may be within one out of four different zones: *normal zone*, *degradation zone*, *excess zone or intolerable zone*. These zones, named measurement zones, are bounded by four values: the minimum and maximum values required for the QoS characteristic (*m* and *M*), and two thresholds values (l_m and L_M), as depicted in Figure 1[4]. We will return to them later, namely to detail how they are defined.

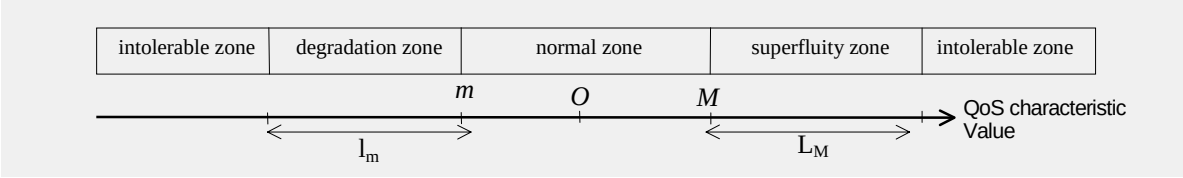

Figure 1 - Measurement zones

For now, it is only important to highlight that the impact of QoS degradation on QoS consumers varies between two fixed values: no impact, *d*; and maximum or near intolerable impact, *D*. This happens when the value of QoS characteristic varies between *m* and m-l_m (see Figure 2). Correspondingly, it is assumed that the impact on communication systems of QoS excess can vary between $-d$ and $-D$[5].
Between those points the measured impact varies linearly, which is also one of the metric assumptions. It is possible to consider different types of variation, for instance a logarithm one, as proposed by Monteiro [Monteiro96, Monteiro95]. That type of variation is, perhaps, better suited to represent the sensitivity of human beings to QoS changes, and thus can be a good choice to regulate the evaluation of its impact. Nevertheless, for now, we decided to use the linear approach in order to avoid complexity. Future work will address other types of sensitivity to changing QoS.
Notice that it is the length of the zones which determines the sensitivity to QoS changes, or the way as the impact to those changes evolves. For instance, the shorter is the degradation zone the bigger will be the *sensitivity* to QoS degradation beyond the minimum *m*. In this case, as shown in Figure 2, smaller changes of the values of the QoS characteristic will induce higher impact variations.
Summing up, the metric uses the concept of measurement zones to evaluate the impact on QoS consumers and suppliers of QoS characteristics deviation from the normal, or the expected, at a given point. As measures refer to the impact of

[3] From now on, unless the contrary is explicitly pointed out, the presentation refers to measures of an individual QoS characteristic (for instance, transit delay, loss, throughput, etc.)
[4] The value *O* represents the QoS characteristic target (or desired) value.
[5] Naturally, it is assumed a null impact when the QoS characteristic assumes the value *O*.

changing QoS on final entities - and not to the absolute values of the QoS characteristics - they are naturally comparable (one of the majors goals of this proposal).
Given that the above, it is necessary to define the measurement zones for each QoS characteristic, and for each point of the communication system where we want to measure. The following sections discuss this fundamental issue.

4. ESTABLISHMENT OF MEASUREMENT ZONES

The establishment of measurement zones will depend on the paradigm used by communication systems to provide QoS. With the *differentiated services* approach, for instance, measurements zones could be pre-defined, reflecting the different QoS to be given to different classes of service. When the supporting mechanism is resource reservation, the applications must specify the characteristics of the traffic they will generate, the flows, and the end-to-end QoS they require. That leads to an agreement between the communication system and the applications – the service contract. As this is clearly the more complex situation for the establishment of measurement zones, from now on we will only consider this case.

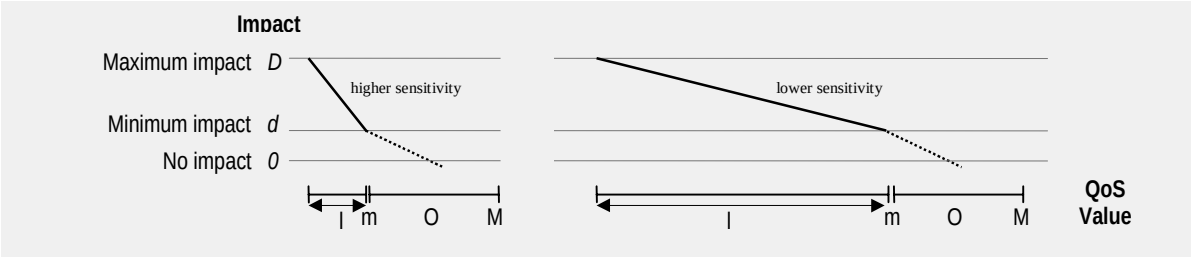

Figure 2 - The degradation zone length, *l*, allows the specification of a higher or lower sensitivity to QoS degradation

Our proposal assumes that the service contract includes, for each flow supported by the communication system, a minimum, a maximum and eventually a target end-to-end value for each QoS characteristic (*m*, *M*, and *O*). The idea is to allow also the specification of the application sensitivity to end-to-end degradation, or the degradation threshold (l_m or l_M, whether the degradation happens before the minimum *m*, or after the maximum *M*, respectively). In order to fully define the measurement zones, it is only lacking the value of the excess threshold, or the communication system sensitivity to excess end-to-end QoS. This zone will provide not only an interesting way to control the traffic generated by applications but also a method to optimize the sharing of communication resources.
To measure the impact of excess QoS we must consider the service supplier point of view. In fact, such a QoS deviation means a less optimized use of the communication resources or a worst communication system availability and performance. It is reasonable to say that this kind of impact depends on the cost of the wasted resources (the ones that should be available for other applications to use). This cost depends on the importance the resources have to applications which, in turn, is related to the degradation thresholds they specify.
Ideally, the communication system should maintain, for each QoS characteristic, historic information on the specified sensitivity to degradation. In this way, it would be possible to calculate the mean recent sensitivity to the degradation of each QoS characteristic, and, from this calculation, the excess thresholds. Nevertheless - and again to avoid complexity - we decided to use a simplifying approach: to define the excess threshold equal to the degradation one, specified in the service contract[6].
Given that the above, all the values necessary to define the measurement zones are determined. They are specified in the service contract and, in the case of the excess threshold, deduced from its values[7]. However, these values have an end-to-end meaning. As we will discuss next, we need more than that in order to turn the metric useful.

1. Impact of the Communication System Subdivision into Modules

We have been discussing end-to-end QoS, more precisely measurement zones that have end-to-end scopes. Those zones are defined by the values established in service contracts, which refer to the performance the traffic should receive from the communication system taken as a whole (from source to destination).

[6] The exploration of other strategies is for further study.
[7] Notice that is possible to simplify the definition of measurement zones. The independent specification of the values *m*, *M*, *O*, l_m or l_M, corresponds to the more generic case and it is not absolutely necessary. It is trivial, for instance, to define some rules for the establishment of measurement zones, based on the types of applications.

By way of illustration, consider a communication system model as the one presented in Figure 3 (left part). Imagine that a data flow, F_i , is established and, for a given QoS characteristic, the following values for limits and thresholds are imposed by the service contract: l_{m_i}, m_i, O_i, M_i, l_{M_i} (with l_{m_i} being the degradation threshold). Those values mean that at point P_r the involved QoS characteristic should vary between *m* and *M*, and that variations beyond these values correspond to QoS degradation or excess – whose impact is measured using the degradation or excess thresholds (l_{m_i} or l_{M_i}). So the referred values are related to the performance expected from the communication system taken as a whole.
However, we want to use a more interesting communication system model than the referred one. Namely, one that considers the communication system as a set of consecutive modules, (Figure 3, right part)[8]. In this case, we should be able to measure the performance of each module taken independently, that is, the QoS at the interface points between them. This implies the "establishment" of measurement zones for those points.
The way used to define measurement zones for the different communication system points has some particularities that must be discussed. The most important one is its strong dependence on the QoS characteristic type. We can grossly divide QoS characteristics into two types: cumulative and non-cumulative characteristics [Monteiro95] as discussed next.

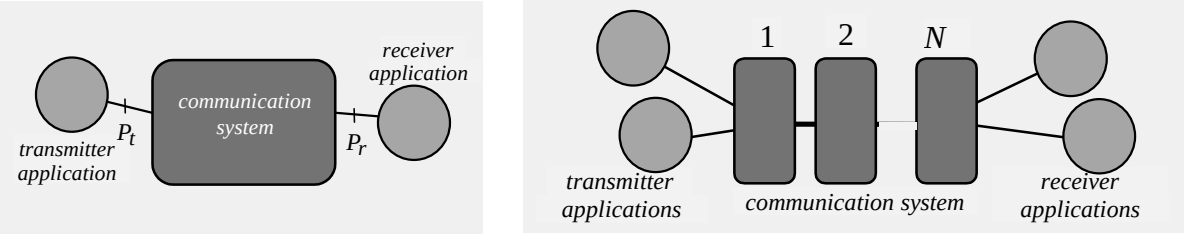

Figure 3 - Communication system models

2. Non-cumulative and cumulative QoS characteristics

Non-cumulative QoS characteristics include the ones for which the final influence of a set of consecutive modules equals the worst influence taken each module separately (as is the case of throughput). When dealing with non-cumulative QoS characteristics, all modules have exactly the same performance responsibilities. The same is to say that the measurement zones to be used at the output of each module do not differ. In fact, the value expected for a non-cumulative QoS characteristic of a given flow is independent of the considered communication system point.
On the other hand, cumulative characteristics include the ones for which the influence of a set of consecutive modules equals the sum of the influence of each module taken separately (as is the case of transit delay). Normally, in this case, different modules have different performance responsibilities, so measurements zones used at their output are usually different.
When cumulative characteristics are being considered, the measurement zones can be used to produce two types of measures: *local* measures and *global* measures, which are discussed next.

Local and Global Measures

Global measures are related to the absolute values expected for the QoS characteristic at a given point. Conversely, local measures are related to the variation that a given module introduces in the QoS characteristic. Measurements of non-cumulative QoS characteristics are always global (it is always possible to know the absolute value this type of QoS characteristic should have at a given point). On the contrary, with cumulative QoS characteristics, measurements could be global or local. Considering, for instance, transit delay, we can clearly see that in order to obtain global measurements a common temporal reference to all the communication system, with enough precision, is needed. In common cases that reference is not available, so local measures, or measures related only to the delay introduced by a given module, must be used. The usefulness of local measures results from their simplicity and the following fact: if all modules respect previously assumed compromises, than the end-to-end characteristic value will be within the expected one.
In order to obtain global measures of cumulative QoS characteristics at a given point *P*, the used measurement zones must reflect the fraction of the end-to-end QoS characteristic that is the responsibility of all the modules from the communication

[8] A module could consist of one of several types of entities. It could be a protocol level, a physical medium or communication equipment, among others. In turn, the communication system subdivision into modules should obey to well defined rules, namely: the modules must be disjoint, the set of all modules must be complete, and the operation sequence of the communications entities must be preserved [Monteiro95].

system input up to point *P*. This could be seen as a fusion of all the modules between those two points in a single virtual module.

Consider the communication system presented in Figure 4, and assume that we are measuring a cumulative QoS characteristic. Each of the three depicted modules will try to assure that a part of the end-to-end characteristic value is not overcome. By other words, to each module is associated a local service contract which defines QoS limits and thresholds values for each QoS characteristic (that are a fraction of the end-to-end ones). In turn, to each virtual module it is possible to associate limits and thresholds equal to the sum of the limits and thresholds that characterize the modules inside it. Given that the above, we can determine the measurement zones in order to quantify the impact of QoS deviations at each module output (points *A*, *B* and *C*). The measures so obtained are global measures, and, notice, they must be weighted with the fraction of the end-to-end QoS characteristic value of the responsibility of the virtual module involved.

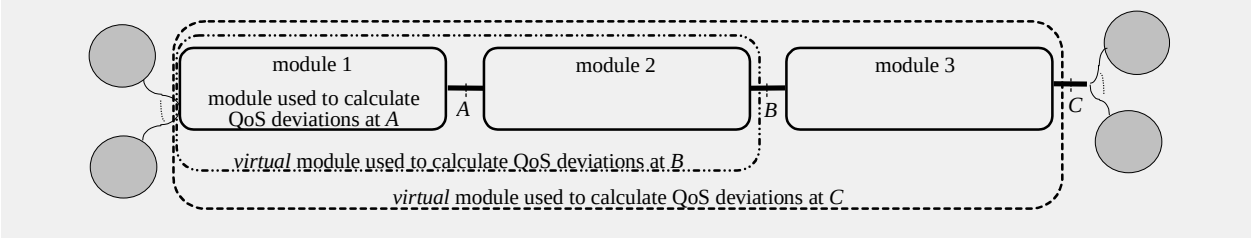

Figure 4 - Consecutive communications modules can be seen as a single virtual one to calculate measurement zones

In order to clarify what has just been said, imagine that we are using a communication system composed of three modules, and that we want to evaluate the transit delay of a certain flow. Suppose, also, that the defined end-to-end limits and the degradation threshold are 80 (*m*), 100 (*M*), and 10 (*lm*)[9]. Finally, suppose that the first module assumes 1/5 of the end-to-end delay, the second also 1/5, and the last one 3/5. The performance responsibility of each module is shown inside them in Figure 5.

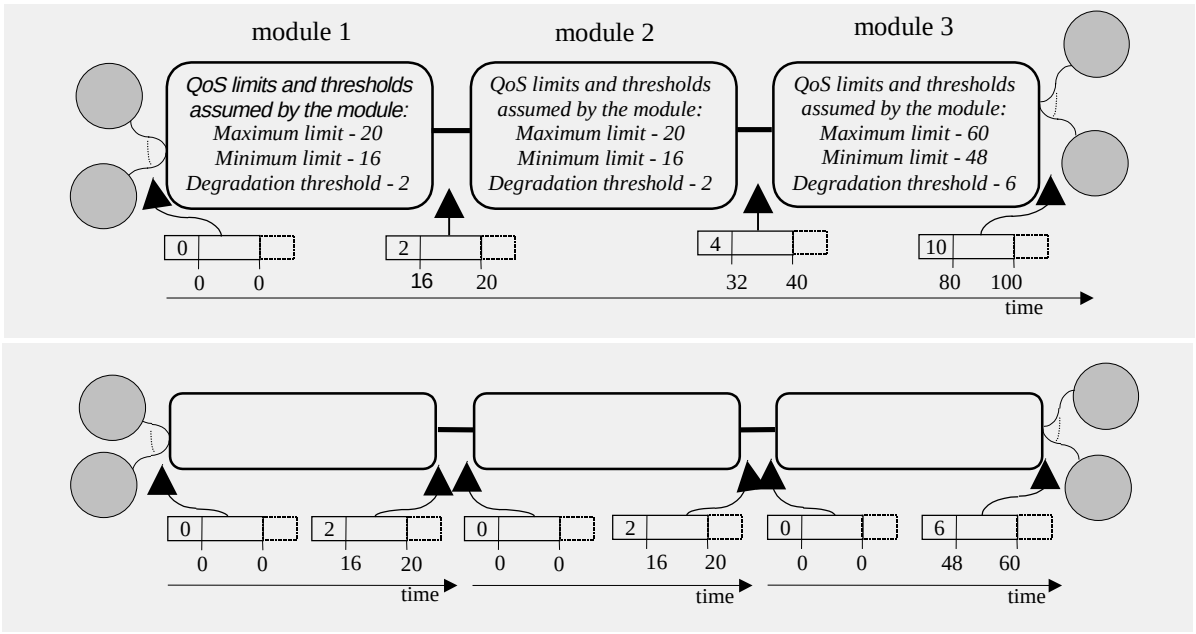

Figure 5 – Top) Measurement zones for the calculation of global measures at different communication system points
Bottom) Measurement zones for the calculation of local measures at different communication system points

[9] We are not considering any specific units, and neither are we considering the excess threshold, because they are useless for the explanation

Adding the correspondent values we can easily obtain the performance responsibilities of a set of contiguous modules, and so the measurement zones to obtain global measures, which are also presented in Figure 5 (top part). Notice that the arriving time of the data to the communication system is assumed as initial temporal reference (thus, all the limits and thresholds used at the communication system input have null values). A different reference could also have been used, for instance to allow policing of the traffic injected into the communication system.
To obtain local measures, the needed measurement zones can be determined considering the temporal reference as the arriving time of the data (that is, not to the communication system input but to the module input). Figure 5 (bottom part) shows, for the example presented before, the measurement zones to be used in this case. Notice that the measured impact must be weighted by the fraction of the end-to-end QoS characteristic value of the module responsibility.
Summing up, the metric basic operation is the quantification - at different communication system points - of the impact of QoS characteristics deviations on applications or communication systems. For this, the metric uses the concept of measurement zones. The referred deviation can have a global meaning, when its evaluation is based on a global reference (the value in fact expected for the characteristic at the considered point) – in this case we are producing global measures. The deviation can also have a local meaning, when it is based on the expected module influence on its value (the expected difference of the characteristic value between module output and module input) – in this case we are producing local measures.

5. TYPES OF QUANTIFICATION

The proposed metric allows two types of quantification:
1) The impact on communication systems or applications of the deviation of QoS characteristics from expected values (for different points of the communication system, for one or various QoS characteristics, and for one flow or a group of flows);
2) The contribution of a communication module, or a set of contiguous modules, to that deviation or impact.
These types of quantification are materialized in two indexes: the *sensitivity index* and the *congestion index* respectively.
The sensitivity index measures the impact of the QoS characteristic deviation (from a certain reference). It takes values from $-D$ to $+D$ and has been implicitly referred before. Figure 6 presents the evolution of the sensitivity index at a communication system point where the measurement zones are characterized by the values: m, M, l_m, and l_M.
The congestion index embodies the evaluation of the performance of communication modules, that is, the module influence on QoS deviation. It is calculated taking the sensitivity indexes at the input and output of the considered module.
Following, we formally present these two indexes related to a given QoS characteristic (of a given flow, naturally).

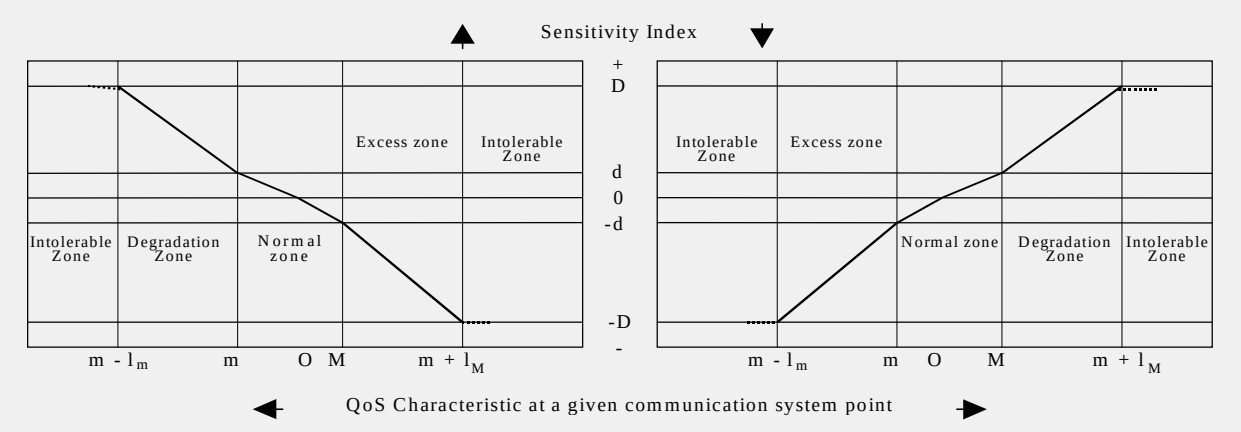

Figure 6 - Sensitivity index evolution at a given communication system point; left part – degradation for values below m (as is the case of throughput); right part – degradation for values above M (as is the case of transit delay)

Definition 1 - *Sensitivity index of a QoS characteristic*

Let q_j be a QoS characteristic of a flow F_i, $v_j(t_k)$ its value at communication system point P ant at instant t_k, and m_j, M_j, O_j, l_{mj}, l_{Mj}, respectively the normal limits, the target value, the degradation and the excess threshold, defined for q_j at P. Then, the sensitivity index of q_j at point P and instant t_k is given by[10]:

[10] If the degradation and excess thresholds were instead l_{m_j} and l_{M_j}, respectively, the index sensitivity would be obtained using the presented formula multiplied by -1.

$$Is^{P}_{Fi,qj}(tk)=\begin{cases}\dfrac{(D-d)(mj-vj(tk))}{l_{mj}}+d & \text{for} \quad -\infty \le vj \le mj \\ \dfrac{Oj-vj(tk)}{Oj-mj} & \text{for} \quad mj \le vj \le 0j \\ \dfrac{Oj-vj(tk)}{Mj-Oj} & \text{for} \quad 0j \le vj \le Mj \\ \dfrac{(D-d)(vj(tk)-Mj)}{l_{Mj}}-d & \text{for} \quad Mj \le vj \le +\infty \end{cases} \tag{1}$$

Definition 2 – *Module congestion index related with a QoS characteristic*

Let q_j be a QoS characteristic of a flow F_i, $Is^{M,In}_{Fi,qj}(tk)$ and $Is^{M,Out}_{Fi,qj}(tk)$ the values of its sensitivity index at module M input and output, at instant t_k. Then, the module M congestion index related to q_j will be given by expression (2) when the characteristic is a non-cumulative one, or expression (3) when the characteristic is a cumulative one[11]:

$$Ic^{M}_{Fi,qj}(tk)=\begin{cases}Is^{M,Out}_{Fi,qj}(tk) & \text{if} \quad Is^{M,In}_{Fi,qj}(tk)\le 0 \vee \left[\left(Is^{M,In}_{Fi,qj}(tk)>0\right)\wedge\left(Is^{M,Out}_{Fi,qj}(tk)>Is^{M,In}_{Fi,qj}(tk)\right)\right] \\ 0 & \text{if} \quad \left(Is^{M,In}_{Fi,qj}(tk)>0\right)\wedge\left(Is^{M,Out}_{Fi,qj}(tk)=Is^{M,In}_{Fi,qj}(tk)\right)\end{cases} \tag{2}$$

$$Ic^{M}_{Fi,qj}(tk)=(Is^{M,Out}_{Fi,qj}-Is^{M,In}_{Fi,qj}(tk))\times\rho_{M} \tag{3}$$

with $\rho_M = \dfrac{O_M}{O_T}$, where O_M is the QoS characteristic target value for module M and O_T the end-to-end one.

Formula (2) states that the congestion index of module M related to a non-cumulative QoS characteristic (such as throughput) equals the correspondent output sensitivity index, except in one case: the QoS characteristic has a worst than normal value at input and maintains this value at output; in this case the congestion index is null. Formula (3) states that module M congestion index related to a cumulative QoS characteristic (such as delay) corresponds to the difference between the output and input sensitivity indexes, weighted by the fraction of the end-to-end value of the characteristic that is due to module M.

Until now, we have been talking about measures related to a single QoS characteristic of a given flow. They are the measures with finest granularity in the proposed metric. As we need to evaluate QoS given to flows taking into account more than one QoS characteristic (for example, delay and losses), it is interesting to have measures which consider several QoS characteristics as a whole. Equally important is the need to evaluate global module performance, which must contemplate all the flows they support. Therefore, we also need measures that consider not only one but several data flows. In our proposal, from the measures related to single QoS characteristics it is possible to obtain measures with larger scope, corresponding to a group of QoS characteristics of a given flow, or to a group of flows.

For a given flow the sensitivity index of a set of characteristics, taken as a whole, can be achieved averaging the sensitivity indexes of each characteristic taken separately, given the *flow sensitivity index*:

$$Is^{P}_{Fi}(t_k)=\frac{1}{n}\cdot\sum_{j=1}^{n}Is^{P}_{Fi,qj}(t_k) \tag{4}$$

[11] Notice that the definition does not contemplate situations where the sensitivity index is worst at input than at output., which is only possible if the module is able to ameliorate the flow's QoS characteristic. This is, nevertheless, an open issue, which can be re-evaluated when considered convenient.

Following the same reasoning, a module congestion index related to several QoS characteristics (of a given flow) can be calculated averaging the module congestion indexes for each characteristic taken independently, at the considered time instant. So, the *module congestion index related to a flow* F_i at the instant of time t_k is:

$$Ic_{F_i}^{M}(t_k) = \frac{1}{n} \cdot \sum_{j=1}^{n} Ic_{F_i,q_j}^{M}(t_k) \quad (5)$$

In turn, if we consider all the flows that a given module supports at a certain instant of time, we can determine the *global module congestion index* at that instant. As before, the global module *M* congestion index can be achieved averaging and weighting:

$$Ic_{g}^{M}(t_k) = \sum_{i=1}^{n} r_i \times Ic_{F_i}^{M}(t_k) \quad (6)$$

This expression uses the concept of *flow relevance* (r_i), which results from the fact that different flows contribute dissimilarly to module congestion, given that they can use distinct module capacities. Assuming that the module resources used by a flow are directly related to the flow bandwidth, flow relevance can be defined as the quotient between the flow bandwidth and the module throughput[12].

Summing up, the metric defines a set of indexes that quantify the impact of QoS deviations from preferred values. It provides a way to measure the QoS provided at a given communication system point, or to measure the performance of a communication system module, related to a QoS characteristic of a flow, a group of QoS characteristics, or a group of flows. All those measures are comparable. This turns possible, for instance, to sort the deviations, or to find out which one is more acute. We argue that if absolute measures were taken we would not have this capacity to sort or to aggregate measures, and so the QoS quantification would be much less useful.

6. METRIC IMPLEMENTATION

This section presents the first approach, developed at the Informatic Engineering Department of the University of Coimbra, for the implementation of the proposed metric . The idea was to construct a prototype to test the metric concepts, analyze its feasibility and evaluate the overhead associate with its use. The prototype was hosted in a FreeBSD operating system mainly for the following reasons:

- as a Unix system FreeBSD provides an easy way to have access to the functionality of a router;
- this operating system grants full access to all its source code, namely to the one which implements the TCP/IP stack;
- an important part of the community doing research in the QoS field is using this system.

The goal of the prototype was to calculate, for a given flow of data, the sensitivity and congestion indexes related with throughput and transit delay (local measures) of a flow. The system is sampled, accordingly to a Poisson distribution as recommended by IPPM [falta aqui a referência]. The different indexes related to the system activity are calculated, which includes the average of the last five values for each index (which gives us a more reliable picture about the very recent past system state).

Figure 7 shows the prototype architecture. In the following we briefly present its different modules.

- Controller Module (M4)
  This module determines when the measures should be taken, or, when the functions of M1 and M2 should be performed. It depends on the MA module operation
- Input Measurements Module (M1)
  This module performs all the measurement activities at network level input. Namely, it timestamps the entering packets and evaluates the volume of data entered in a precise measured window of time. It depends on the M3 module operation.
- Output Measurements Module (M2)
  The same as M1 module, for outgoing packets.
- Index Evaluation Module (M4)
  After the measures are calculated, by M1 and M2 modules, this module calculates the sensitivity and congestion indexes.
- Random Number Generator Module (MA) and QoS State Visualization module (MB)
  Auxiliary modules, responsible for generate random numbers and for showing to users the system status related with QoS, namely the value of the recently measured indexes. The MB module corresponds to a modification made to Unix *netstat* program).

[12] Another approach is to let the system to define the *flow relevance* for each flow. If the used communication system allows priorities between flows, this could be an interesting alternative.

Main decisions and challenges related with the prototype development

The main challenge connected with the development of the prototype was a consequence of the FreeBSD characteristics related with time. As this is not a real-time operating system, we had some problems to guarantee, with enough precision, that certain activities will occur in the due instant of time – for instance the sampling activities. The consequence is that we need to improve system awareness about the precise time, which can easily result in a huge operation overhead and consequently poor performance. We used the function microtime() to obtain the exact information about time (wich gives precision of 1μs). As this function consumes too much CPU cycles, we are changing the code to directly read (using embedded assembler instructions) the time-stamp counter available in Intel Pentium® processors. With this approach, we expect to obtain much better performances and higher time resolutions.

Some of the most important decisions made during the development of the prototype were decided with a main goal in mind: to improve efficiency and scalability. Some examples are:

- the packet arrival time at the IP level is stored in the free space available in the system mbuf which contains it. This avoids the waste of memory space and, most important, the need to search it when the value is needed (as is the case if it was stored in specific data structures);
- the QoS evaluation is based on system sampling (as mentioned before);
- a random number generator was constructed. To avoid the impact of its operation on system performance, the numbers are pre-generated by a process running in user space and stored in a table constructed in system space. From time to time those numbers are refreshed. Whenever the system needs a random number it will get it from the table;
- part of the activities related to delay and throughput measurements are made (and must be made) in critical zones of the kernel code – as, for instance, the macros which manipulate system queues; nevertheless, the indexes are not calculated in those zones; this can result in some latency to obtain the indexes, which we assume negligible, but minimizes the risk of losing the execution of important system tasks;
- all the changes to kernel code were made inside the IP level; therefore the prototype can be experimented with different types of logical layer stuff.

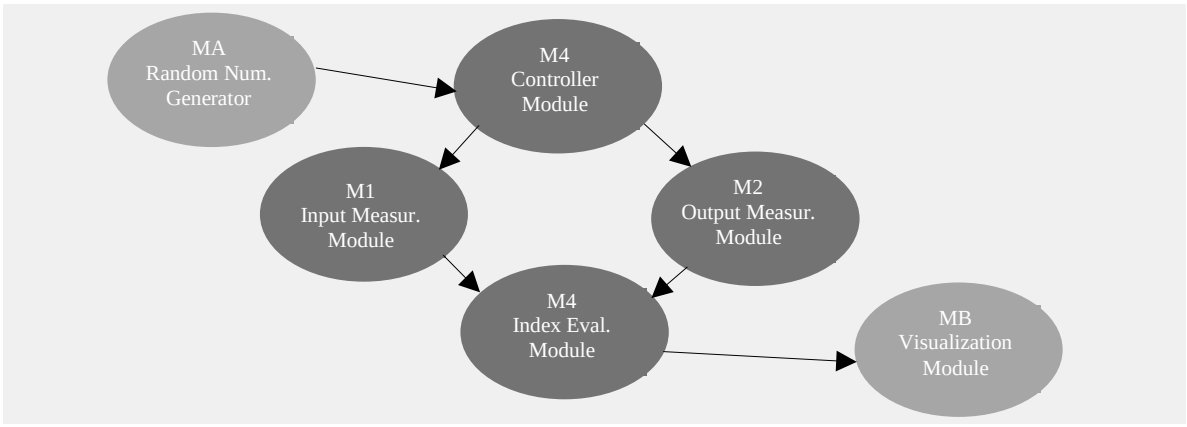

Figure 7 – Prototype architecture [Magalhaes98]

Performance tests

After the prototype implementation, we submitted it to a diverse set of tests. The idea was to analyze its robustness, correctness, and performance. We will now focus our attention on the performance tests made.

The main goal of the performance tests was to quantify the overhead associated with the metric operation. For this purpose, we used a simple test-bed: two hosts (*calvin* and *obelix*) interconnected by a third one (*ideafix*), performing router functions. We then compared the performance of the router running normal FreeBSD code (v. 2.1.5) with its performance running the kernel code modified to include our metric.

These comparisons were made with the router submitted to different load conditions. We used the MGEN [MGEN] application to generate flows of 64 bytes packets, at different rates, from host *calvin* to host *obelix*. In this way, we simulated the following scenarios: no load, loads of 256Kbps, 512Kbps, 768Kbps, and 1024Kbps.

In turn, the ping command was used to evaluate the router performance. For each load situation, and for routing running normal and modified kernels, we executed the ping command 4 times[13]. In each time, 500 packets were generated. In the first experience packets had a payload of 64 bytes and in the second one payloads of 1000 bytes.

[13] More precisely, the command *ping obelix* was executed in *calvin*.

The mechanism to sample the system was modified specifically for the tests. During tests, QoS was measured periodically, during 30 ms time windows separated 150 ms from each other. The numeric test results are shown in Table 1. These are shown graphically in Figure 8 (where it is also depicted the overhead associated with the metric operation).

Load (kbps)	RTT (ms) – normal kernel Min / avg / max / stddev		RTT (ms) – kernel with metric Min / avg / max / stddev	
0	1-	4.235 / 4.278 / 13.891 / 0.433	1-	0.669 / 0.683 / 0.857 / 0.433
	2-	4.235 / 4.257 / 4.453 / 0.032	2-	0.666 / 0.681 / 0.852 / 0.032
	3-	4.235 / 4.258 / 4.629 / 0.038	5-	0.666 / 0.681 / 1.564 / 0.038
	4-	4.235 / 4.256 / 4.481 / 0.031	6-	0.668 / 0.681 / 0.844 / 0.031
	avg	4.235 / 4.262 / 6.863 / 0.031	avg	0.667 / 0.725 / 3.432 / 0.265
256	1-	0.652 / 1.952 / 9.543 / 1.493	1-	0.643 / 1.843 / 9.314 / 1.405
	2-	0.652 / 2.050 / 12.322 / 1.716	2-	0.645 / 1.737 / 12.474 / 1.212
	3-	0.654 / 1.761 / 4.136 / 1.127	3-	0.643 / 1.727 / 4.458 / 1.117
	4-	0.651 / 1.763 / 4.056 / 1.125	4-	0.644 / 1.930 / 36.809 / 2.341
	avg	0.652 / 1.881 / 7.514 / 1.365	avg	0.643 / 1.809 / 15.763 / 1.518
512	1-	0.637 / 3.629 / 14.115 / 3.019	1-	0.633 / 3.112 / 10.253 / 2.481
	2-	0.644 / 3.371 / 10.746 / 2.829	2-	0.637 / 3.095 / 10.554 / 2.511
	3-	0.643 / 3.437 / 11.384 / 2.894	3-	0.636 / 3.085 / 23.006 / 2.581
	4-	0.646 / 3.508 / 22.910 / 2.960	4-	0.636 / 3.037 / 9.655 / 2.415
	avg	0.643 / 3.486 / 14.788 / 2.925	avg	0.635 / 3.082 / 13.367 / 2.497
768	1-	0.654 / 4.642 / 11.494 / 4.021	1-	0.632 / 4.113 / 12.310 / 4.351
	2-	0.653 / 4.513 / 11.155 / 3.923	2-	0.619 / 4.019 / 20.981 / 3.986
	3-	0.653 / 4.683 / 11.198 / 4.093	3-	0.608 / 4.029 / 22.671 / 4.816
	4-	0.653 / 4.707 / 10.910 / 4.106	4-	0.612 / 4.047 / 29.681 / 4.299
	avg	0.653 / 4.636 / 11.189 / 4.035	avg	0.617 / 4.047 / 21.410 / 4.363
1024	1-	0.674 / 7.733 / 64.875 / 5.446	1-	0.671 / 6.889 / 60.910 / 5.312
	2-	0.678 / 7.495 / 25.042 / 4.541	2-	0.669 / 6.498 / 22.187 / 4.542
	3-	0.734 / 7.750 / 87.379 / 5.905	3-	0.701 / 6.639 / 31.023 / 4.544
	4-	0.725 / 7.558 / 20.754 / 4.522	4-	0.723 / 6.559 / 10.398 / 4.601
	avg	0.702 / 7.634 / 49.512 / 5.103	avg	0.691 / 6.646 / 31.129 / 4.749

Table 1 – Results of the overhead tests [Magalhaes98]

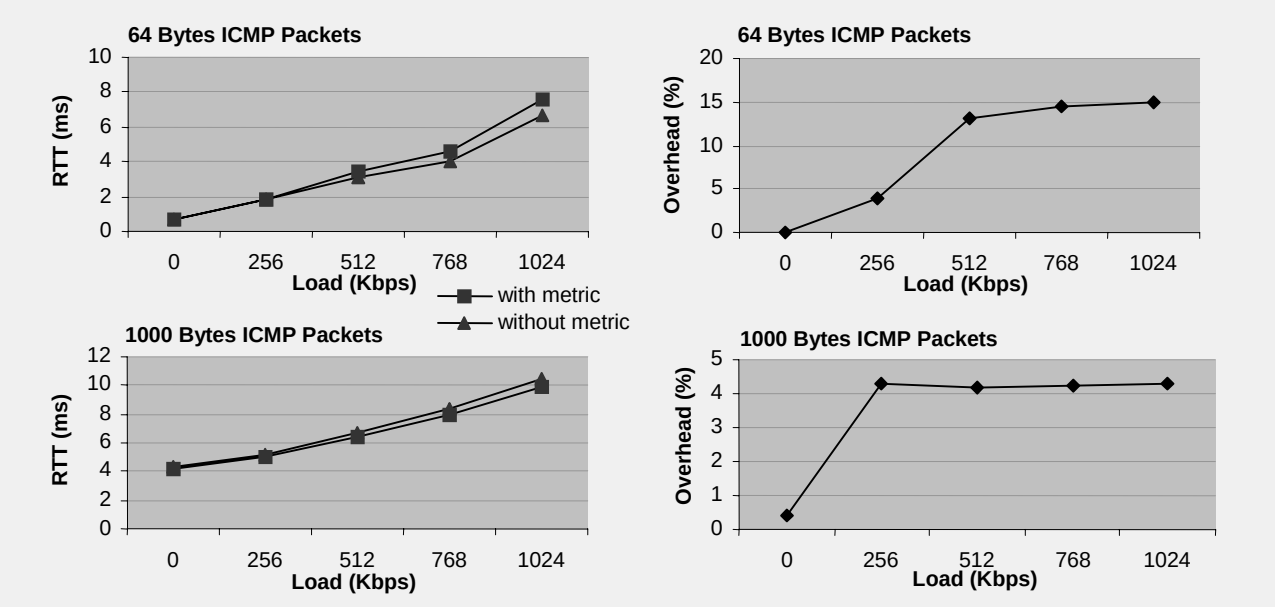

Figure 8 – Left part: evolution of the average RTT of the 2000 probe packets (4*500) with load;
Right part: evolution of the metric overhead with load (overhead = fraction of the average RTT using normal kernel that was added to the average RTT when a kernel with the metric was used).

As expected, the overhead grows with load and is greater for flows based on smaller packets. Nevertheless, its value should be lower, which is especially evident for flows made of small packets, with rates starting a little above 256Kbps. We have identified the main reason for this poor result[14], and we are now working to overcome it or to reduce its magnitude. Besides, notice that we can always adjust the metric operation overhead changing the sampling rate of the system. The mean time between samples is an important trade-off that should be carefully considered for proper metric operation: a lower sample rate means lower metric overhead but also lower system responsiveness to changes in communications system status.
Also notice that if we were running the metric in a real-time platform the problems which are lowering system performance would naturally disappear. Thus, without any optimization work, the prototype performance would be much better.

7. CONCLUSION AND FUTURE WORK

This paper highlighted the lack of proposals to measure QoS and the negative impact that this fact has on the construction of QoS-capable communication systems. It emphasized also the importance of a metric to measure QoS, the main difficulties in its conception and, in our opinion, the characteristics it should have.
From there, a proposal to measure QoS in packet switched networks is presented. We argue that the proposed metric will facilitate the development of an effective and truly integrated QoS management system, which we consider very important to construct efficient QoS-capable communication systems.
Last, this paper presents the first approach developed at the Informatics Engineering Department of the University of Coimbra to construct the proposed metric. The prototype has been constructed in a FreeBSD system, and allowed to test the metric basic concepts, to assess its feasibility, and to measure its associated overhead. The main decisions and challenges related to the construction of the prototype were discussed, as well the results of the overhead tests made to it.
Presently we are testing some solutions to substantially decrease the metric overhead and we are beginning a new project to develop a differentiated services capable router[DS]. Our plan is to use the proposed metric to give different QoS to different classes of traffic. In doing so, as we are continually evaluating the QoS actually given to the traffic classes, we expect to dynamically adjust the priority given to their respective packets, and thus to achieve better communication resource sharing and utilisation.

8. ACKNOWLEDGEMENTS

We would like to highlight the work produced by Maria Arminda Lopes and Ana Julia Magalhaes, namely their participation in the development of the prototype and in the overhead tests that were made to it.

9. REFERENCES

[Partridge93] Craig Partridge, “Protocols for high-speed networks: some questions and a few answers”, *Computer Networks and ISDN Systems*, Vol. 25, pp. 1019-1028, 1993.

[Steinmetz97] R. Steinmetz, L. C. Wolf , "Quality of Service: Where are We", In *IWQOS’97 Proceedings*, New York, USA, May 1997.

[ISOIEC9680] ISO/IEC JTC1/SC21 N9680, Open Systems Interconnection, Data Management and Open Distributed Processing, Information Technology - Quality of Service - Framework, Final CD, July 95.

[Monteiro96] Edmundo Monteiro, Gonçalo Quadros, Fernando Boavida, "A Scheme for the Quantification of Congestion in Communication Services and Systems", In *Proceedings of the third International Workshop on Services in Distributed and Networked Envirnoments*, IEEE ComSoc, Macau, 3-4 June 96.

[Monteiro95] Edmundo Monteiro, *Controlo de Congestão em Sistemas Intermediários de Camada de Rede*, (Congestion Control in Network Layer Intermediate Systems) Ph.D. Thesis, University of Coimbra, Portugal, 95. (portuguese)

[]IPPM] http://io.advanced.org/IPPM/

[Magalhaes98] Ana Magalhaes, Arminda Lopes, *Medidor de Qualidade de Serviço em Redes IP*, (QoS monitoring in IP networks), Technical Report, Department of Informatics Engineering of the University of Coimbra, Portugal, April 98. (portuguese)

[DS] http://www.ietf.org/html.charters/diffserv-charter.html

[MGEN] http://manimac.itd.nrl.navy.mil/MGEN/

[14] The precise time must be known with high frequency. As mentioned before, this is done using the *micotime()* function.