Quality of Experience Provision in the Future Internet

C. Bruni, F. Delli Priscoli, G. Koch, A. Palo, and A. Pietrabissa

***Abstract*—This work deals with the satisfaction of the quality of experience (QoE) requirements in the perspective of the emerging future Internet framework. The evolution of the Internet is pointing out its limitations, which are likely to hinder its potential. In this respect, this paper introduces an innovative approach to cope with some key limitations of the present communication networks. In particular, the need of efficiently utilizing the available network resources and of guaranteeing the user expectations in terms of QoE requires a full cognitive approach, which is realized by the introduction of a novel architecture design, the so-called future Internet core platform. The future Internet core platform aims at bringing together the applications world with the network world, hence introducing a further cognitive level while enabling a new generation of applications: network-aware applications. This paper is concerned with an important aspect of the intelligent connectivity between applications and network: the service class association, which, if performed with a cognitive approach, can yield some important improvements and advantages in the emerging information era. The key idea presented in this paper is a real-time dynamic control procedure for the selection of the optimal service class. The approach is based on theoretical considerations validated by a proof-of-concept simulation.**

***Index Terms*—Cognitive networks, future Internet, network-aware applications, optimal control, quality of experience (QoE).**

I. Introduction

FUTURE Internet design is one of the current priorities established by the European Union (UE). FI-WARE FP7 [1] is a major UE project which is currently trying to address the issues raised by such design. In this respect, this paper presents some concepts which can contribute to evolve the issues which are being dealt with in this project: nevertheless, being the work still in progress, the ideas included in this paper are not necessarily the ones of the FI-WARE consortium. In particular, the optimal control procedure presented in this paper has been developed in the framework of the PLATINO National Project [2] activities by further developing and elaborating the concepts that emerged in the FI-WARE project.

In the authors' vision, the future Internet overall target is to allow applications to transparently, efficiently, and flexibly exploit the available network resources, aimed at achieving a satisfaction level meeting the personalized users' needs and expectations [3]. Such expectations could be expressed in terms of a properly defined quality of experience (QoE) [4], [5], which could be regarded as a personalized function of quality of service (QoS), security, mobility, . . ., parameters (we note that the QoE concept is becoming widespread in all the emerging network paradigms; e.g., QoE approaches are being proposed in Content Delivery Networks [6], [7], and in Cloud networks [8]).

The future Internet aim is to satisfy the personalized user expectations for plenty of applications that are being developed. In fact, in the near future, it is expected that the application requirements can be personalized with respect to the user and even with respect to the specific application instance: this concept will be referred to as personalized QoE application requirements. In this respect, the traditional approach of statically mapping applications on a limited set of service classes guaranteeing predefined performance is no longer satisfactory [9], [10], as detailed later.

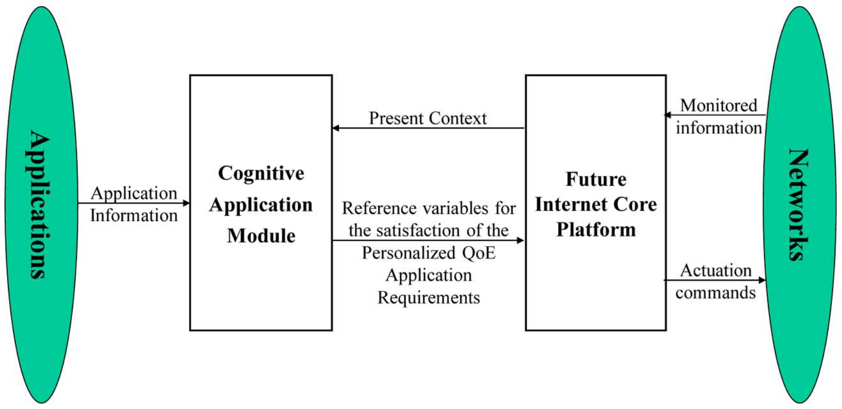
Fig. 1 highlights the high-level future Internet reference scenario [11]. The cognitive application module interacts with the specific application protocols in order to deduce technology-independent personalized QoE application requirements. The cognitive application module is in charge of driving the future Internet core platform to satisfy the requirements, based on selected feedback information provided by the future Internet core network—the so-called present context. Note that the application module is referred to as "cognitive" just because it bases its derivations on feedback information, namely, the present context.

The future Internet core platform consists of a set of cooperative technology-independent algorithms and procedures, which are referred to as "network control functionalities" in the following discussion. These, possibly based on selected feedback information provided by proper sensing functionalities monitoring the networks and on the reference variables provided by the cognitive application module, are in charge of taking control decisions concerning specific network control problems (e.g., resource management, security management, mobility management, service/content management, etc.). These decisions are enforced on the networks by proper actuation commands.

It should be clear that the proposed cognitive application module can be developed independently of the network control functionalities, i.e., it can be used in conjunction with any type of such functionalities (either cognitive or not), and these last can continue to operate according to their usual way of working. In other words, owing to the cognitive application module, the whole future Internet core platform becomes closed loop (i.e., cognitive) regardless of the actual way of working of the network control functionalities (these last can be either open or closed loop).

Manuscript received November 26, 2013; revised April 3, 2014 and June 4, 2014; accepted July 22, 201. Date of publication September 19, 2014; date of current version February 23, 2016. This work was supported in part by the EU project FI-WARE under FP7 Grant Agreement 619520, by the Italian project PLATINO under Grant Agreement PON01_01007, and by the EU project FI-WARE under Grant Agreement 285248.

The authors are with the Department of Computer, Control and Management Engineering, University of Rome "La Sapienza," 00185 Rome, Italy (e-mail: brunic@dis.uniroma1.it; dellipriscoli@dis.uniroma1.it; san_lorenzo@libero.it; palo@dis.uniroma1.it; pietrabissa@dis.uniroma1.it).



Applications
Application Information
Cognitive Application Module
Present Context
Reference variables for the satisfaction of the Personalized QoE Application Requirements
Future Internet Core Platform
Monitored information
Actuation commands
Networks

Fig. 1. Future Internet concept.

The presented future Internet core platform concept can be realized by means of a distributed framework consisting of appropriate agents to be transparently embedded in properly selected network nodes (e.g., mobile terminals, base stations, backhaul network entities, and core network entities). The multiagent organization; the design of the relevant algorithms, procedures, and interfaces; and the mapping of these agents into selected existing network nodes and their transparent embedding in such nodes are all very challenging issues that future Internet designers have to cope with. These problems have to be addressed environment by environment, also taking into account the requirement of a smooth transition (future Internet functionalities are expected to be gradually embedded into the network nodes).

In particular, this paper focuses on the cognitive application module which, according to the aforementioned approach, should deduce the reference variables driving the cognitive network control functionalities toward the satisfaction of the personalized QoE application requirements. Instead, the cognitive network control functionalities are outside the scope of this paper; instances of such functionalities can be found in [12]–[17].

In particular, this paper deals with the problem of associating each application, which has its peculiar own requirements as the QoE is considered (i.e., the personalized QoE application requirements) to the most appropriate service class among the few ones (with respect to the number of possible different applications) supported by the network. The main innovations of the proposed algorithm are the following.

1) It performs the choice of the service class dynamically, based on real-time measures of the network traffic, whereas current approaches assign the applications to a given service class at the start of the application.
2) It aims at improving the QoE experienced by the users by fulfilling the personalized QoE application requirements and not, as usually pursued by the network resource management algorithms, at fulfilling the QoS requirements of the applications.

This paper is organized as follows. Section II includes some key definitions and outlines the proposed cognitive application module architecture. Section III presents the modeling of the cognitive application module. Section IV includes the problem formulation and the related solution. Section V presents some simulation results. Finally, Section VI draws the conclusion.

II. Basic Definitions and Concepts

By *microflow*, we mean the flow of packets relevant to a given in-progress application instance, relevant to the same source and the same destination, and having specific personalized QoE application requirements. In general, a given application instance is supported by one or more microflows: for instance, a given bidirectional teleconference taking place in a given network, between two terminals A and B, is supported by 4 microflows, i.e., two audio microflows (from A to B and vice versa) and two video microflows (from A to B and vice versa). Nevertheless, in the following, for the sake of clarity and without loss of generality, we will consider application instances supported by one microflow (the general case can be simply handled by foreseeing an application agent, as defined in the following discussion, for each microflow), so we will just refer to applications instead of to microflows. In addition, for the sake of brevity, whenever confusion is not possible, when mentioning “application” we mean “application instance.”

In the present implementations, each application $A^{(i)}$, $i = 1, 2, \ldots, m$, is *statically* associated (statically means for the whole application duration) to a *service class* $u^{(i)}$, properly selected in the set $\{1, 2, \ldots, N\}$ of predefined service classes at application setup. A given *service class* $j (j = 1, 2, .., N)$ is characterized by a set of QoS requirements making reference to proper parameters (e.g., minimum service availability, minimum throughput, etc.), which have to drive the network control functionalities included in the future Internet core platform. As a matter of fact, such network control functionalities should assure the satisfaction of the service class requirements associated to the parameters in question (e.g., the service availability should be greater than the minimum one, the experienced throughput should be greater than the minimum one, etc.).

It should be evident that this static association is not suitable for coping with the ever-growing number of applications with personalized QoE requirements in the future Internet era [18]. This is due to, on the one hand, the limited number of service classes against the more and more increasing number of application types and, on the other hand, the fact that the existing service class requirements do not, in general, match with the personalized QoE application requirements.

In addition, in the static approach, the knowledge of the appropriate service class is needed before the application starts its execution. This means that the static association requires an *a priori* knowledge of the way the association in question has to be carried out. As a matter of fact, at present, the most common adopted approach is that the application declares, at its setup, in some implicit way, the type of traffic it is willing to transport over the network. Then, the service end-point has to somehow figure out the best service class it should statically associate to the application flow. A possible method is the classification proposed in [9]. Nevertheless, this entails difficulties in continuously updating the static association between application types and service classes, since in the current future Internet scenario, new applications with personalized QoE requirements are being created every day [19], [20].

A further consideration to be taken into account concerns possible unexpected variations in time of network conditions; this can especially happen in mobile scenarios, e.g., due to sudden shadowing/fading phenomena and/or to traffic peaks. Currently, in general, the selected service class corresponds to a specific scheduling policy and/or modulation scheme and/or frequency band, etc., tailored on a standard network behavior. Therefore, the static association in question prevents the real-time adaptation of the selected service class on sudden unexpected variations of network conditions.

In the light of the aforementioned discussions, it should be clear that the static association between applications and service classes entails the following limitations.

1) Due to the very different personalized QoE application requirements, in general, a given application does not perfectly match any service class.
2) The *a priori* knowledge of the appropriate service class might pose serious limitations in the current future Internet scenario in which new applications with personalized QoE requirements are being created every day.
3) The optimal association should be varied in real time since it depends on the present network conditions which could be subject to sudden variations (e.g., due to traffic dynamics).

In order to overcome the aforementioned drawbacks, this paper proposes an innovative *dynamic* association between applications and service classes. The idea is that the cognitive application module, on the basis of the personalized QoE application requirements and of the present context, is in charge of selecting the most appropriate service class $u^{(i)}(t_k)$ at each time t_k, for each application $A^{(i)}$, where $t_k,\ k = 1, 2, \ldots$, denotes the time instants at which network control is enforced. The present context should include properly selected feedback information accounting, for instance, for the network present traffic situation. The key criterion underlying the aforementioned dynamical selection is to approach, as far as possible, a performance level meeting the various personalized QoE application requirements.

In the literature, a large variety of models for QoE computing and associated QoE requirement is being proposed [21]. Several proposed QoE models represent the QoE as a simple function of one or more QoS parameters (*objective* QoE). For example, in [22], the so-called IQX hypothesis is proposed, with an exponential relation between the QoE and the most significant QoS parameter. In [23], a QoE model is developed, which quantifies the QoE level as a convex combination of quantities depending on QoS parameters (delay, jitter, and packet loss ratio) and on the service parameters (interruption ratio, access success ratio, and access response time). In [24], the QoE level of an application of a given service class is computed by a nonlinear monotonically increasing function, named correlation model; three service classes are considered (guaranteed, premium, and best effort services), and therefore, three correlation models are proposed. Other approaches consider also user feedback (*subjective* QoE) in the QoE computation (see [25] and references therein). In this respect, the approach followed in this paper is extremely flexible since it leaves completely open the QoE definition and the associated QoE requirements, thus allowing its tailoring to the specific applications.

According to the aforementioned approach, we propose to define the personalized QoE application requirements by providing the following.

1) A vector $s^{(i)}(t_k) \in \mathbb{R}^s$ of selected feedback variables, representing the present context of the application $A^{(i)}$ at time t_k, which impact the QoE experienced by the application $A^{(i)}$.
2) A function $\varphi^{(i,k)}$ allowing us to measure the present QoE experienced by the application $A^{(i)}$ at time t_k, on the basis of the sets $s^{(i)}(t_k), \ldots, s^{(i)}(t_{k-M})$ of selected reference variables; the function $\varphi^{(i,k)}$ is defined so that it assumes values in the range [0,1].
3) A *QoE reference*, hereinafter indicated as $\overline{Q}^{(i)}(t_k)$, which is the reference value for the personalized QoE application $A^{(i)}$ requirements; even the QoE reference $\overline{Q}^{(i)}(t_k)$ is defined so that it assumes values in the range [0,1].

Now, we can define the following *QoE error function*:

$$e^{(i,k)} = \overline{Q}^{(i)}(t_k) - \varphi^{(i,k)}\left(s^{(i)}(t_k), \ldots, s(i)(t_{k-M})\right). \quad (2.1)$$

If the aforementioned error is *positive*, at time t_k, the application $A^{(i)}$ is experiencing a nonsatisfactory QoE (*underperforming application*). If the aforementioned error is *negative*, at time t_k, the application $A^{(i)}$ is experiencing a QoE even better than expected (*overperforming application*). Note that this last situation is desirable only if the network is idle; conversely, if the network is congested, the fact that a given application $A^{(i)}$ is overperforming is not, in general, desirable since it may happen that such application is subtracting valuable resources to other applications which are underperforming.

Note that the aforementioned approach has the key advantage of leaving completely open the QoE definition, so a proper selection of $s^{(i)}(t_k)$, $\varphi^{(i,k)}$, and $\overline{Q}^{(i)}(t_k)$ can allow us to tailor the QoE to the characteristics of the considered network and of the considered application.

In the light of the aforementioned discussion, the cognitive application module should *dynamically* determine the most appropriate service class $u^{(i)}(t_k)$ aimed at avoiding, as far as possible, the occurrence of underperforming applications; in case this is not possible (e.g., due to an overall network congestion), the dynamical selection in question should aim at guaranteeing fairness among the QoE errors relevant to the in-progress applications.

III. Modeling of the Cognitive Application Module

The proposed cognitive application module is organized according to a number of *application agents*: each in-progress application $A^{(i)}$ has its own application agent $C^{(i)}$ which is in charge of interworking with the application protocols in order to deduce the function $\varphi^{(i,k)}$ and the QoE reference $\overline{Q}^{(i)}(t_k)$ characterizing the personalized QoE application $A^{(i)}$ requirements.

The application agent $C^{(i)}$ is in charge of dynamically selecting, at each time t_k, the most appropriate service class $u^{(i)}(t_k)$ which should be associated to the application $A^{(i)}$; this selection is based on the following input information:

1) the selected local feedback variables $s^{(i)}(t_k), \ldots, s^{(i)}(t_{k-M})$ representing the present context of the application $A^{(i)}$ at times $t_k, \ldots, t_{k-M}$;
2) the already mentioned information ($\varphi^{(i,k)}$, $\overline{Q}^{(i)}(t_k)$) derived from the interaction with the application protocol;
3) a global coordination signal, namely, the so-called *status signal* $\Pi(t_k)$ accounting for the present overall network status at time t_k, broadcast by a single *supervisor agent* (SA; see the following discussion for further details).

As already pointed out in the introduction, the aforementioned agents have to be carefully embedded in properly selected network nodes (e.g., base stations, mobile terminals, etc.). A key criterion for mapping these agents into the network nodes is to avoid overwhelming the considered network with signaling overhead. For instance, a possible criterion aimed at saving signaling overhead is to colocate the application agents with the monitoring functionalities in charge of deriving the feedback parameters. Therefore, in general, the application agents relevant to different applications have to be embedded in different network nodes.

Then, the desirable requirement that no signaling exchange takes place among application agents is imposed by the following: 1) the requirement of saving signaling overhead; 2) the fact that the application agents will be, in general, embedded in different network nodes; and 3) the fact that the number of applications (and hence of application gents) simultaneously in progress is constantly increasing (the considered future Internet architecture has to scale up to the Internet of Things). Therefore, these agents should take their decisions independently from one another, without exchanging information.

Clearly, this issue entails a very complex problem of *coordination* among the application agents of the considered network. Indeed, a given application agent $C^{(i)}$ has to take real-time decisions relevant to the most appropriate service class $u^{(i)}(t_k)$ without knowing the decisions taken by the other application agents. In this respect, note that, as a given application agent $C^{(i)}$ selects a given service class $u^{(i)}(t_k)$, such selection impacts the performance of the other applications. In fact, the network control functionalities aim at handling the network resources so that, as far as possible, all service class requirements are simultaneously satisfied.

The proposed approach for maintaining a certain coordination among application agents without any signaling exchange among these agents is to foresee a single SA. The SA is in charge of monitoring the global behavior (possibly by sampling), at any time t_k, of the selected feedback variables $s^{(i)}(t_k)$, $i = 1, 2, \ldots, m$, where m represents the number of in-progress applications, and of broadcasting a status signal including the relevant information to all application agents.

Therefore, the information coming from the status signal is a further important input for the application agents, helping them to take consistent decisions, partially compensating the fact that they do not exchange signaling information to one another. Of course, the status signal broadcasting entails the presence of a certain signaling overhead; nevertheless, the amount of such overhead can be kept rather limited, owing to the following.

1) The signaling communication only occurs from the SA to the application agents, i.e., one-to-many (no communication is foreseen in the opposite direction).
2) In the proposed approach, the status signal only includes network status information at service class level (and not at application level), which entails a reasonable small amount of information in consideration of the limited number of service classes.

In the proposed approach, the SA has just to properly collect, elaborate, and broadcast appropriate measurement parameters. This means that the SA is a *passive* equipment in the sense that it does not take part in the decision process relevant to the service class selection, which is completely demanded to the algorithm embedded in the application agents.

Remark 3.1: The SA is part of the network control plane and can be physically embedded in already existing network nodes acting as network controllers. For example, in the emerging software-defined network (SDN) paradigm [26], the control plane is decoupled from the data plane and is realized by a centralized controller, which drives the network hardware; in these networks, the SA functions may be embedded in the SDN controller, and the SA control messages may be conveyed to the network hardware together with the SDN messages. Other examples are the base stations in wireless access networks or the cluster-head nodes in *ad hoc* networks: these nodes, which act as network controllers, may be seamlessly enriched with the SA functionalities.

In the light of the aforementioned discussion, the resulting cognitive application module is modeled as shown in Fig. 2.

Fig. 2 shows that the cognitive application module includes, for each application $A^{(i)}$, $i = 1, 2, \ldots, m$, an application agent $C^{(i)}$, with the task of choosing the class of service $u^{(i)}(t_{k+1})$, $k = 0, 1, 2, \ldots$, for the application itself at time t_{k+1} and of communicating it to the core platform. Each agent $C^{(i)}$ performs this task on the basis of the following.

1) The state $s^{(i)}(t_k)$ of the application $A^{(i)}$ at time t_k, $k = 1, 2, \ldots$, as well as of the last M values $\{s^{(i)}(t_{k-h}), h = 1, 2, \ldots, M\}$, for a fixed memory length $M > 0$. For

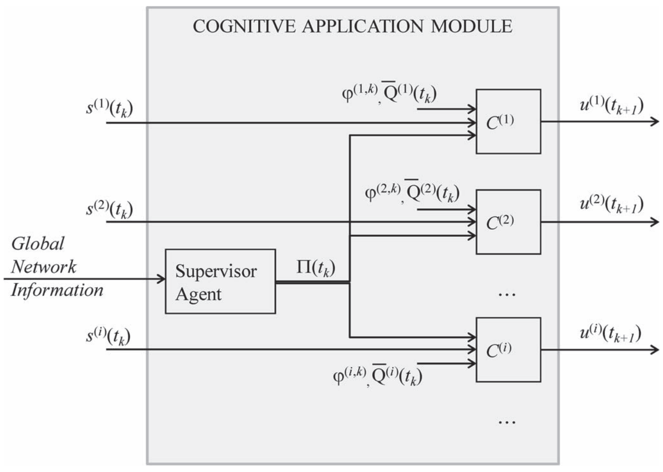

COGNITIVE APPLICATION MODULE
$s^{(1)}(t_k)$
$s^{(2)}(t_k)$
$s^{(i)}(t_k)$
Global Network Information
Supervisor Agent
$\Pi(t_k)$
$\varphi^{(1,k)}, \overline{Q}^{(1)}(t_k)$
$\varphi^{(2,k)}, \overline{Q}^{(2)}(t_k)$
$\varphi^{(i,k)}, \overline{Q}^{(i)}(t_k)$
$C^{(1)}$
$C^{(2)}$
$C^{(i)}$
$u^{(1)}(t_{k+1})$
$u^{(2)}(t_{k+1})$
$u^{(i)}(t_{k+1})$
...
...

Fig. 2. Cognitive application module model.

notation brevity, we introduce the vector $z^{(i)}(t_k) \in \mathbf{R}^{S(M+1)}$

$$z^{(i)}(t_k) = \begin{pmatrix} s^{(i)}(t_k) \\ s^{(i)}(t_{k-1}) \\ \dots \\ s^{(i)}(t_{k-M}) \end{pmatrix}. \tag{3.1}$$

2) The function $\varphi^{(i,k)}$, in charge of the computation of the QoE $Q^{(i)}(t_k)$ from $z^{(i)}(t_k)$

$$Q^{(i)}(t_k) = \varphi^{(i,k)}\left(z^{(i)}(t_k)\right) \tag{3.2}$$

along with the reference value $\bar{Q}^{(i)}(t_k)$ for the QoE itself.

3) The current value of the status signal components $\Pi^{(j)}(t_k) \in \mathbb{R}^V, j = 1, 2, \dots N$, which includes the V parameters that we will use (see the next section) to describe the statistical distribution of the S selected feedback variables. For convenience, we introduce also the vector $\Pi(t_k) \in \mathbf{R}^{NV}$ representing the whole status signal

$$\Pi(t_k) = \begin{pmatrix} \Pi^{(1)}(t_k) \\ \Pi^{(2)}(t_k) \\ \dots \\ \Pi^{(N)}(t_k) \end{pmatrix}. \tag{3.3}$$

IV. Choice of the Service Classes

As mentioned in the previous section, at time t_k, each application agent $C^{(i)}$ has the task of autonomously deciding the service class $u^{(i)}(t_{k+1})$ to be assigned to its application $A^{(i)}$ at time t_{k+1}. This paper proposes to formalize the task as an optimization problem. This can be appropriately done by seeking an optimization policy related to a given cost function $J^{(i)}(u(t_{k+1}))$. The definition of $J^{(i)}(u)$ should account both for the goal of the good cost of $A^{(i)}$ and for the need of an efficient balanced utilization of the network resources.

A meaningful structure for $J^{(i)}(u)$ appears to be the following:

$$\begin{aligned} &J^{(i)}\left(u(t_{k+1})\right) \\ &= E_{u(t_{k+1})}\left\{\left(\bar{Q}^{(i)}(t_{k+1}) - \varphi^{(i,k+1)}\left(z^{(i)}(t_{k+1})\right)\right)^2 \middle| z^{(i)}(t_k)\right\} \end{aligned} \tag{4.1}$$

where E_u denotes the mean value under control $u = u(t_{k+1})$; the mean value (with respect to $z^{(i)}(t_{k+1})$) is estimated by means of the statistical distribution of the feedback variables $\Pi(t_k)$.

Note that in (4.1) the mean value is conditioned on the information available up to instant t_k carried by the already known values $z^{(i)}(t_k)$. Thus, the only uncertainty is referred to the next value $s^{(i)}(t_{k+1})$, which is needed to evaluate the function $\varphi^{(i,k+1)}(z^{(i)}(t_{k+1}))$ [see (3.1) and (3.2)].

Therefore, the function $\varphi^{(i,k+1)}(z^{(i)}(t_{k+1}))$ may be conveniently substituted by a function just depending on the value of $s^{(i)}(t_{k+1})$

$$\begin{aligned} &J^{(i)}\left(u(t_{k+1})\right) \\ &\quad = E_{u(t_{k+1})}\left\{\left(\bar{Q}^{(i)}(t_{k+1}) - \tilde{\varphi}^{(i,k+1)}\left(s^{(i)}(t_{k+1})\right)\right)^2\right\} \\ &\quad = E_{u(t_{k+1})}\left\{\left(e^{(i,k+1)}\right)^2\right\} \end{aligned} \tag{4.2}$$

where $e^{(i,k+1)}$ is the error at time t_{k+1} as defined in (2.1).

To compute (4.2), we now need a (forecast) model for what the behavior of $s^{(i)}(t_{k+1})$ would be, given $z^{(i)}(t_k)$, under any possible choice of $u(t_{k+1})$.

We here propose a simple model, in which $s^{(i)}(t_{k+1})$, under service class u, is given by a convex combination of $s^{(i)}(t_k)$ and the mean value

$$\mu^{(u)}(t_k) = E_u\left[s(t_k)\right] \tag{4.3}$$

where $s(t_k)$ denotes the state variable of an application with service class u at time t_k. To that convex combination, we add an uncertainty term modeled as an additive Gaussian noise $\nu(t_k)$, with zero mean and covariance matrix

$$\Psi_\nu^{(u)}(t_k) = E_u\left[\left(s(t_k) - \mu^{(u)}(t_k)\right)\left(s(t_k) - \mu^{(u)}(t_k)\right)^{\mathrm{T}}\right] \tag{4.4}$$

(where the superscript T denotes the transpose operator).

Then, the forecast model is represented as

$$s^{(i)}(t_{k+1}) = \alpha^{(u)}(t_k)s^{(i)}(t_k) + \left(1 - \alpha^{(u)}(t_k)\right)\mu^{(u)}(t_k) + \nu(t_k). \tag{4.5}$$

The coefficient $\alpha^{(u)}(t_k)$ in (4.5) has to lie in [0,1]. A possible choice is

$$\alpha^{(u)}(t_k) = \frac{\sqrt{\mathrm{Tr}\left\{\Psi_\nu^{(u)}(t_k)\right\}}}{\max\limits_{\theta=1,.,\dots,N}\sqrt{\mathrm{Tr}\left\{\Psi_\nu^{(\theta)}(t_k)\right\}}}. \tag{4.6}$$

We first note that the quantities (4.3) and (4.4) [and therefore (4.6)] are available for the agent $C^{(i)}$ at time t_k since they are typically included in $\Pi(t_k)$. The choice (4.6) corresponds to attributing more weight to the current state value $s^{(i)}(t_k)$ in (4.5) whenever $Tr\{\Psi_\nu^{(u)}(t_k)\}$ is higher (i.e., $\mu^{(u)}(t_k)$ is a poorly reliable forecast value).

We are now able to compute (4.2).

Indeed, under any choice $u^{(i)}(t_{k+1})$, we know that the probability density $p_{u^{(i)}}(t_{k+1})^{(i)}$ of $s^{(i)}(t_{k+1})$ enjoys a Gaussian structure with mean vector

$$\bar{s}^{(i)}\left(u^{(i)}(t_{k+1})\right) = \alpha^{\left(u^{(i)}(t_{k+1})\right)}(t_k)s^{(i)}(t_k) + \left(1-\alpha^{\left(u^{(i)}(t_{k+1})\right)}\right)\mu^{\left(u^{(i)}(t_{k+1})\right)}(t_k) \quad (4.7)$$

and covariance matrix $\Psi_\nu^{(u^{(i)}(t_{k+1}))}(t_k)$.

Thus, we have to compute the cost function at time t_{k+1} as

$$J^{(i)}\left(u^{(i)}(t_{k+1})\right) = \int\limits_{R^S}\left(\bar{Q}^{(i)}(t_{k+1}) - \tilde{\varphi}^{(i,k+1)}(s)\right)^2 p^{(i)}_{u^{(i)}(t_{k+1})}(s)ds. \quad (4.8)$$

The optimal choice of the service class at time t_{k+1} for the application $A^{(i)}$ may then be accomplished by computing (4.8) for each $u^{(i)}(t_{k+1})$, $i = 1, 2, \ldots, N$, and looking for the minimum value.

In general, for the sake of implementation simplicity, the integral at the RHS in (4.8) may be analytically computed, at the price of introducing some approximations. For instance, two possible approximations are the following.

1) If we assume a polynomial structure for $\tilde{\phi}$, in the neighborhood of $\bar{s}^{(i)}(u^{(i)}(t_{k+1}))$, then (4.8) enjoys an explicit form depending on $\mu^{(u^{(i)}(t_{k+1}))}(t_k)$ and $\Psi_\nu^{(u^{(i)}(t_{k+1}))}(t_k)$. In particular, for an affine structure

$$\tilde{\varphi}^{(i,k+1)}(s) = a^{(i,k+1)}(t_k) + b^{(i,k+1)\mathrm{T}}(t_k)s \quad (4.9)$$

we have

$$\begin{aligned}J^{(i)}&\left(u^{(i)}(t_{k+1})\right)\\ &= E_{u^{(i)}(t_{k+1})}\left\{\left(a^{(i,k+1)}(t_k) + b^{(i,k+1)\mathrm{T}}(t_k)s(t_k) - \bar{Q}^{(i)}(t_{k+1})\right)^2\right\}\\ &= b^{(i,k+1)\mathrm{T}}(t_k)\Psi_\nu^{\left(u^{(i)}(t_{k+1})\right)}(t_k)b^{(i,k+1)}(t_k)\\ &\quad + \left(a^{(i,k+1)}(t_k) + b^{(i,k+1)\mathrm{T}}(t_k)\bar{s}^{(i)}\left(u^{(i)}(t_{k+1})\right) - \bar{Q}^{(i)}(t_{k+1})\right)^2. \quad (4.10)\end{aligned}$$

2) If $\Psi_\nu^{(u^{(i)}(t_{k+1}))}$ is sufficiently small, then we may approximate the error $e^{(i,k+1)}$ as

$$e^{(i,k+1)} \approx \bar{Q}^{(i)}(t_{k+1}) - \tilde{\varphi}^{(i,k+1)}\left(\bar{s}^{(i)}\left(u^{(i)}(t_{k+1})\right)\right). \quad (4.11)$$

Then, $J^{(i)}(u^{(i)}(t_{k+1}))$ is approximated as follows:

$$J^{(i)}\left(u^{(i)}(t_{k+1})\right) \approx \left(\bar{Q}^{(i)}(t_{k+1}) - \tilde{\varphi}^{(i,k+1)}\left(\bar{s}^{(i)}\left(u^{(i)}(t_{k+1})\right)\right)\right)^2. \quad (4.12)$$

Then, in this case, the statistical distribution of each selected feedback variable $s^{(i)}(t_k)$ can be described by just a single value [namely, the mean value yielded by (4.3)], i.e., $S = V$, for each selected feedback variable.

Remark 4.1: The communication overhead of the proposed approach is kept limited, as it appears from the following facts: 1) no communication is required among the agents; 2) no communication is required from the agents to the SA; and 3) the SA messages are broadcast to the agents, by using the same control channels already existing in the specific network (see *Remark* 3.1) and, whenever possible, by exploiting the broadcast capabilities of the specific networks. Furthermore, the amount of information in the status signal is small: in case the approximation of (4.12) is used, since $S = V$, the status signal is a vector having NS components.

Remark 4.2: By using the approximation of (4.12) and by using a simple QoE function $\tilde{\phi}$, it is clear that the algorithm complexity of the control algorithm is negligible. In fact, at each time instant t_k, the controller $C^{(i)}$ of each agent i is in charge of computing the values of the cost function $J^{(i)}(u^{(i)}(t_{k+1}))$ for each possible choice of $u^{(i)}(t_{k+1})$ (i.e., for each possible service class) and then of choosing the action which returns the minimum value. The complexity in computing (4.12) depends on the complexity of the QoE function, which, typically, is very low [22]–[24].[1]

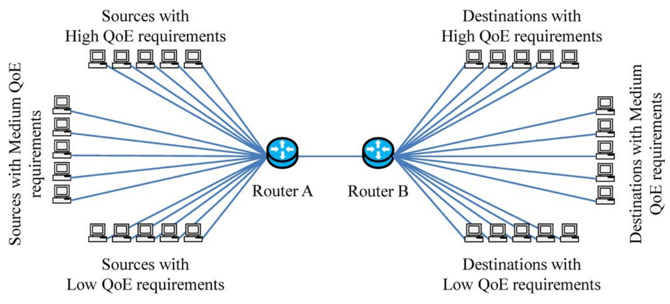
V. Simulations

In order to provide a proof-of-concept of the proposed dynamic approach, up to the authors' knowledge, the proposed approach is the first one which dynamically varies the association of applications to the available service classes; therefore, a simulation environment has been setup, aimed at testing whether the proposed dynamic control procedure helps in overcoming the limitations of the static control approach (described in Section II).

The performed simulations have been carried out by using OPNET Modeler [28], which is a powerful network simulation tool allowing us to model communication systems and to predict network performances. The chosen simulation scenario is very simple and is the classical dumbbell network topology (widely used to test fairness protocols). Indeed, our purpose is not to simulate a real network but to provide a first attempt to test the actual way of working of the proposed approach. The purpose of the simulations is to show that the innovative architecture presented in this paper, together with the proposed dynamic association between service classes and applications, can yield real improvements in performance with respect to the static association adopted so far.

Fig. 3 shows the simulated scenario. There are 15 application instances (i.e., $m = 15$); each application instance entails a monodirectional traffic exchange from 15 data sources (the nodes on the left) which generate the traffic to 15 correspondent data destinations (the nodes on the right).

[1] In this respect, we note that the QoE model complexity is unlikely to impair the proposed approach. For instance, [8] and [27] consider genetic algorithm and machine learning approaches to infer the QoE level from both network and subjective parameters: computational resources are required only in the (offline) training phase of the model, which is not part of the proposed approach.



Sources with High QoE requirements
Destinations with High QoE requirements
Sources with Medium QoE requirements
Destinations with Medium QoE requirements
Router A
Router B
Sources with Low QoE requirements
Destinations with Low QoE requirements

Fig. 3. Simulation scenario.

The 15 application instances belong to three *application types*: two classes of video conferencing and one File Transfer Protocol. In the following, for the sake of brevity, these three application types will be referred to as *high*, *medium*, and *low*, respectively. The application instances belonging to the same application type are characterized by the same value (*high*, *medium*, and *low*, respectively) of the QoE reference. There are five application instances running for each application type, where each application instance is associated in a one-to-one fashion to a different source–destination pair. Router A multiplexes the traffic, coming from the different sources, over the single link present between routers A and B; then, router B demultiplexes the aggregated traffic toward the corresponding destination links.

The application agents and the SA have been implemented in the C programming language and have been embedded in all of the destination nodes and in router A, respectively. In router A, very simple network control functionalities are also embedded, namely, a standard network control procedure, with four queues corresponding to the four considered service classes (i.e., $N = 4$) scheduled according to a weighted fair queuing mechanism.

In the considered simulation scenario, the static control procedure foresees that the four service classes are permanently associated to the three application types. This means that an application instance belonging to a given application type is associated for the whole application lifetime to the same service class. Conversely, the dynamic control procedure foresees that each application agent dynamically selects, even during application lifetime, the most appropriate service class to be associated to the relevant application instance in order to "drive" the network control functionalities toward the minimization of the performance index (4.1).

In the simulated scenario, we have considered three selected feedback variables (i.e., $S = 3$):

1) the transfer delay [s] $D^{(i)}(t_k)$ which, for the traffic relevant to the application i at time t_k, is the average delay experienced by its packets from the time they exit from the source node to the time t_k when they arrive at the destination node;
2) the admitted bit rate [b/s] $R^{(i)}_{\text{adm}}(t_k)$ which, for the traffic relevant to the application i at time t_k, is the bit rate of traffic admitted (by the network control procedures) to be carried from the source node to the destination node;
3) the loss bit rate [b/s] $R^{(i)}_{\text{loss}}(t_k)$ which, for the traffic relevant to the application i at time t_k, is the bit rate of traffic lost in the run from the source node to the destination node.

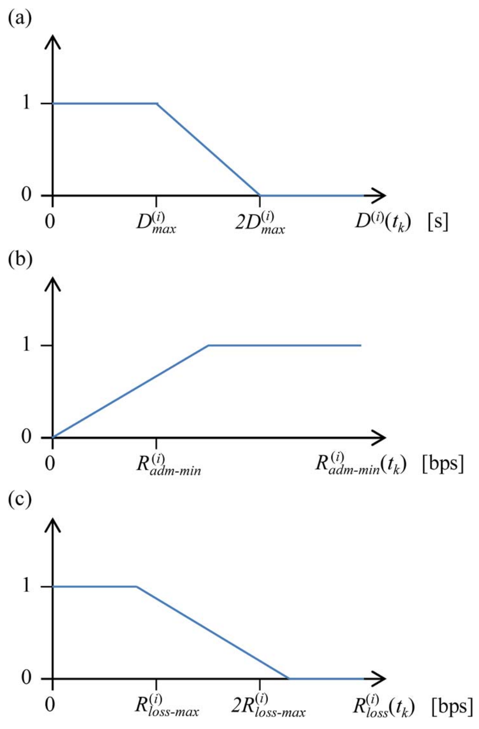


Fig. 4. Feedback variable functions involved in the QoE evaluation.

As far as the function $\varphi^{(i,k)}$ for QoE evaluation is concerned, the performed simulations consider approximation 2 discussed in Section IV. In particular, we have assumed the following simple function based on a suitable convex combination of functions of the aforementioned feedback variables:

$$\begin{aligned}
&\tilde{\varphi}^{(i,k+1)}\left(\bar{s}^{(i)}\left(u^{(i)}(t_{k+1})\right)\right)\\
&= \alpha_1 \max\left\{0, \min\left[1, 1 - \frac{D^{(i)}(t_k) - D^{(i)}_{\max}}{D^{(i)}_{\max}}\right]\right\}\\
&\quad + \alpha_2 \max\left\{0, \min\left[1, 1 - \frac{R^{(i)}_{\text{adm}-\min} - R^{(i)}_{\text{adm}}(t_k)}{R^{(i)}_{\text{adm}-\min}}\right]\right\}\\
&\quad + \alpha_3 \max\left\{0, \min\left[1, 1 - \frac{R^{(i)}_{\text{loss}}(t_k) - R^{(i)}_{\text{loss}-\max}}{R^{(i)}_{\text{loss}-\max}}\right]\right\}
\end{aligned} \tag{5.1}$$

where the α_1, α_2, and α_3 parameters are nonnegative with the constraint $\alpha_1 + \alpha_2 + \alpha_3 = 1$ (so that $\tilde{\phi}^{(i,k+1)}(\bar{s}^{(i)}(u^{(i)}(t_{k+1})))$ takes values in the range [0, 1]), and the following thresholds have been considered:

1) $D^{(i)}_{\max}$: *Maximum transfer guaranteed delay* [s] relevant to the application i;
2) $R^{(i)}_{\text{adm}-\min}$: *Minimum guaranteed bit rate* [b/s] relevant to the application i;
3) $R^{(i)}_{\text{loss}-\max}$: *Maximum guaranteed loss bit rate* [b/s] relevant to the application i.

The rationale behind the three terms that are summed up in expression (5.1) can be easily understood by plotting them, as shown in Fig. 4.

As in the dynamic control procedure, each application agent autonomously and dynamically decides, at each decision instant t_k (the decision interval $t_k - t_{k-1}$ has been set equal to 5 s), according to the procedure described in Sections III and IV, the service class minimizing the cost index (4.1). In this last respect, the performed simulations consider approximation 2 discussed in Section IV, which leads to the cost index $J^{(i)}(u(t_{k+1}))$ yielded by the expression (4.12). This means that, in the simulated scenario, the statistical distribution of each selected feedback variable $s^{(i)}(t_k)$ can be described by just a single value (i.e., $S = V = 3$) for each service class: this is the mean value, yielded by (4.3), allowing us to compute $\bar{s}^{(i)}(u^{(i)}(t_{k+1}))$. Then, the status signal $\Pi(t_k)$ is a vector having 12 components.

At each decision instant t_k, the QoE measured by each application agent has been computed both in the static and dynamic control procedures. The comparison between the two approaches has been carried out with respect to a cost index obtained by averaging (4.12) with respect to the 15 considered application instances and to the considered decision time intervals occurring during the simulation run. Thus, the cost index in question is calculated with the following expression:

$$\bar{J}(u) = \frac{\sum_{i=1}^{15} \sum_{k=1}^{P} \left(\tilde{\varphi}^{(i,k)} \left(\bar{s}^{(i)}(t_k) \right) - \bar{Q}_{t_k}^{(i)} \right)^2}{15 * P} \tag{5.2}$$

where P denotes the total number of decision intervals occurring in the considered simulation run. Obviously, the lower the cost index (5.2) is, the lower the average square error is, between the measured QoE and the reference one.

The simulation scenario presented so far has been tested in three different simulation setups aimed at testing whether the proposed dynamic control procedure helps in overcoming limitations 1–3 presented in Section II of the static control procedure. The comparisons between the static and dynamic approaches have been carried out by using the cost index (5.2).

The first simulation setup focuses on limitation 3, by considering sudden degradations occurring over network links, i.e., variable network conditions. Instead, in such simulation setup, limitations 1 and 2 are not considered: this means that, in the static case, an optimal requirement matching between application types and service classes has been assumed. The comparison is carried out for two different congestion levels of the network: *light* and *severe*. The congestion levels are obtained by varying the available capacity of the link that connects router A with router B while maintaining the same average throughput of all of the applications' flows. In this respect, the *light* and *severe* congestion levels are obtained by setting up, respectively, 0.90 and 1.15 ratios between the average offered traffic and the capacity of the bottlenecked link. Table I summarizes the simulation results.

The second simulation setup focuses on limitations 1 and 2, by considering a uniformly distributed random initial assignment of service classes to application instances. In the static case, such initial assignment is kept for the whole application lifetime, while in the dynamic case, it is varied according to the dynamic control procedure. In this simulation setup, limitation 3 is not considered: this means that network conditions do not vary during all simulation runs. Table II summarizes the simulation results.

TABLE I
COST INDEX (5.2) IN VARIABLE NETWORK CONDITIONS

	Static Control Procedure	Dynamic Control Procedure
Congestion *Light*	0.0328	0.0199
Congestion *Severe*	0.0333	0.0217

TABLE II
COST INDEX (5.2) IN CASE OF RANDOM INITIAL ASSIGNMENT OF APPLICATION INSTANCES TO SERVICE CLASSES

	Static Control Procedure	Dynamic Control Procedure
Congestion *Light*	0.0383	0.0174
Congestion *Severe*	0.0490	0.0204

TABLE III
COST INDEX (5.2) IN THE FAVORABLE SETUP FOR THE STATIC CASE

	Static Control Procedure	Dynamic Control Procedure
Congestion *Light*	0.0062	0.0067
Congestion *Severe*	0.0095	0.0109

The third simulation setup does not consider any of the three limitations (1–3) presented in Section II of the static control procedure (*optimal setup for the static case*). This means that, in the static case, an optimal requirement matching between application types and service classes has been assumed, and network conditions are stationary. It should be clear that, in this simulation setup, the static control procedure is expected to achieve a better performance with respect to the dynamic one. As a matter of fact, on the one hand, no limitation is considered among the ones which have motivated the introduction of the dynamic control procedure; on the other hand, the static control procedure can benefit from the optimal requirement matching between application types and service classes. Table III summarizes the simulation results.

The simulation results reported in Tables I and II show a remarkable performance advantage of the dynamic control procedure with respect to the static one, as soon as the phenomena related to limitations 1–3, presented in Section II, are taken into account. This advantage is particularly evident when the congestion level of the network is severe, due to the ability in a fast rearranging of application instance requirements, which is inherent in the dynamic control approach.

Moreover, the results reported in Table III show that, even when the aforementioned limitations are not considered, the dynamic control approach does not exhibit a significant disadvantage with respect to the favorable setup for the static case.

Fig. 5 depicts the QoE, evaluated with the formula (5.1), for three different application instances belonging to high, medium, and low application types utilizing the dynamic control procedure. The dashed lines represent the QoE reference values of the three application types. The figure highlights the fact that the QoE of the different applications tracks the respective QoE reference values.

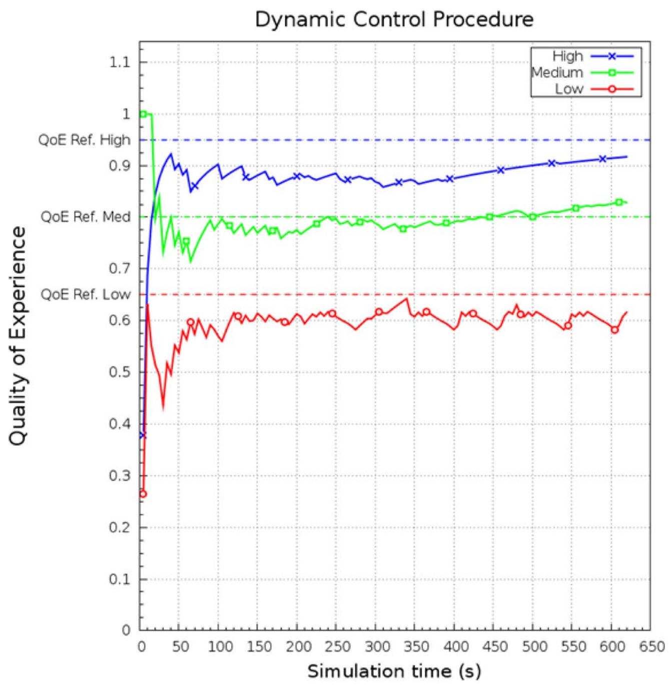


Fig. 5. Measured QoE for application instances characterized by high, medium, and low values for the reference QoE.

VI. Conclusion

Conceptually, this paper can be thought of as composed of two parts. The key idea that connects the two parts is the innovative core concept of the dynamic assignment of the service class.

The first part of this paper (Sections I–III) highlights some key objectives characterizing the future Internet core platform design, which are being developed with the contribution of the authors in the framework of the EU FI-WARE project. The future Internet concept is presented, with a particular focus on the cognitive application module. This last introduces an innovative cognitive approach in the communication paradigm, aimed at optimizing the utilization of network resources by adopting intelligent procedures for the satisfaction of the personalized QoE application requirements.

The cognitive application module relies on, but is decoupled from, the network control functionalities (i.e., they might be either closed or open loop). Therefore, it can be utilized both on emerging and future network technologies but also on well-established legacy and proprietary networks. Moreover, the distributed solution of the cognitive application module, together with the centralized SA, makes the proposed approach scalable and flexible without overwhelming the network with signaling information.

The proposed approach has the key advantage of leaving the QoE definition completely open. Thus, a proper selection of the feedback variables, QoE reference, and function of QoE measurement can allow us to tailor the QoE to the characteristics of the considered network and of the considered application. The mechanism used to enforce the cognitive control decisions is the dynamic (in time) selection of the service class associated to the application.

The second part of this paper (Sections IV and V), based on the work performed by the authors in the framework of the PLATINO National Project, describes a possible optimal control procedure that can be adopted in order to dynamically decide the optimal choice of the service class. The procedure defines an appropriate optimization policy and a cost function aimed, on the one hand, at guaranteeing the personalized QoE application requirements and, on the other hand, at assuring an efficient and balanced utilization of network resources.

A simple reference implementation of the dynamic control procedure is developed in OPNET Modeler, just to provide a proof-of-concept of the proposed approach. Several simulation runs were carried out, in different network congestion levels and conditions, in order to check the correct way of working of the proposed procedure. In particular, the simulations show that the resulting measured QoE actually tracks the reference QoE and also highlight the advantages that the proposed dynamic control approach enjoys compared to the static one.

The theoretical considerations, validated through the simulations, seem to highlight several advantages of the proposed procedure, which allow us to overcome limitations 1–3 identified in Section II.

1) The resulting QoE levels can be fine grained at will, in contrast to the finite discrete levels (equal to the limited number of classes of service that the network offers) provided by the static control procedure.
2) No *a priori* knowledge of the characteristics of a new application is needed, for the dynamic assignment of the appropriate service class.
3) The expected user requirements in terms of QoE are satisfied, even in nonstationary network conditions.

The authors would like to note that this work intends to present a preliminary but encouraging step toward the formalization of the QoE concept, as well as the design of the QoE architectural framework and of the control procedure, allowing us to efficiently and fairly satisfy the personalized QoE application requirements. They are well aware that the work presented in this paper needs further theoretical insights, as well as much more complete validation in realistic scenarios.

Acknowledgment

The authors would like to thank all of the members of the FI-WARE and PLATINO consortium, particularly Prof. C. Gori Giorgi, Prof. S. Monaco, and Prof. S. Battilotti of the University of Rome "La Sapienza"; nevertheless, the concepts and the results contained in this paper are not necessarily the ones of the consortiums.

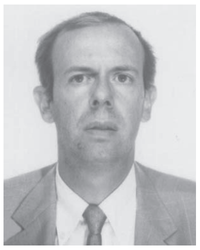
References

[1] *FI-WARE (Future Internet Core Platform) EU FP7 Project*, Contract no. 285248. [Online]. Available: www.fi-ware.eu
[2] *PLATINO (PLATform for INnOvative Services in Future Internet).* [Online]. Available: http://www.progettoplatino.it/

[3] M. Castrucci, F. Delli Priscoli, A. Pietrabissa, and V. Suraci, “A cognitive future Internet architecture,” in *The Future Internet*, vol. 6656, *Lecture Notes in Computer Science*. Berlin, Germany: Springer-Verlag, 2011, pp. 91–102.
[4] R. Jain, “Quality of experience,” *IEEE Multimedia*, vol. 11, no. 1, pp. 96–95, Jan.–Mar. 2004.
[5] M. Fiedler, T. Hossfeld, and P. Tran-Gia, “A generic quantitative relationship between quality of experience and quality of service,” *IEEE Netw.*, vol. 24, no. 2, pp. 36–41, Mar./Apr. 2010.
[6] S. Manfredi, F. Oliviero, and S. P. Romano, “A distributed control law for load balancing in content delivery networks,” *IEEE/ACM Trans. Netw.*, vol. 21, no. 1, pp. 55–68, Feb. 2013.
[7] F. Z. Yousaf, M. Liebsch, A. Maeder, and S. Schmid, “Mobile CDN enhancements for QoE-improved content delivery in mobile operator networks,” *IEEE Netw.*, vol. 27, no. 2, pp. 14–21, Mar./Apr. 2013.
[8] W.-H. Hsu and C.-H. Lo, “QoS/QoE mapping and adjustment model in the cloud-based multimedia infrastructure,” *IEEE Syst. J.*, vol. 8, no. 1, pp. 247–255, Mar. 2014.
[9] T. Karagiannias, K. Papagiannaki, and M. Faloutsos, “BLINC: Multilevel traffic classification in the dark,” in *Proc. SIGCOMM*, Philadelphia, PA, USA, Aug. 21–26, 2005, pp. 229–240.
[10] C. Estan, S. Savage, and G. Varghese, “Automatically inferring patterns of resource consumption in network traffic,” in *Proc. SIGCOMM*, 2003, pp. 137–148.
[11] M. Castrucci *et al.*, “Key concepts for the future Internet architecture,” in *Proc. Future Netw. Mobile Summit*, Warsaw, Poland, Jun. 2011, pp. 1–10.
[12] A. Pietrabissa, “Optimal call admission and call dropping control in links with variable capacity,” *Eur. J. Control*, vol. 15, no. 1, pp. 56–67, 2009.
[13] A. Pietrabissa, “A reinforcement learning approach to call admission and call dropping control in links with variable capacity,” *Eur. J. Control*, vol. 17, no. 1, pp. 89–103, 2011.
[14] A. Pietrabissa, “An alternative LP formulation of the admission control problem in multi-class networks,” *IEEE Trans. Autom. Control*, vol. 53, no. 3, pp. 839–845, Apr. 2008.
[15] A. Pietrabissa, F. Delli Priscoli, A. Fiaschetti, and F. Di Paolo, “A robust congestion control for communication networks with time-varying delays,” in *Proc. IEEE CACSD Int. Conf. Control Appl.*, Munich, Germany, Oct. 4–6, 2006, pp. 2093–2098.
[16] F. Delli Priscoli and A. Pietrabissa, “Load-adaptive bandwidth-on-demand protocol for satellite networks,” in *Proc. 41st IEEE CDC*, Las Vegas, NV, USA, Dec. 10–13, 2002, pp. 4066–4071.
[17] R. Cusani, F. Delli Priscoli, G. Ferrari, and M. Torregiani, “A novel MAC and scheduling strategy to guarantee QoS for the new-generation windflex wireless LAN,” *IEEE Wireless Commun.*, vol. 9, no. 3, pp. 46–56, Jun. 2002.
[18] P. Reichl, “From ‘quality-of-service’ and ‘quality-of-design’ to ‘quality-of-experience’: A holistic view on future interactive telecommunication services,” in *Proc. 15th Int. Conf. Softw., Telecommun. Comput. Netw.*, Sep. 2007, pp. 1–6.
[19] A. Moore and K. Papagiannaki, “Toward the accurate identification of network applications,” in *Proc. PAM*, Mar. 2005, pp. 41–54.
[20] Y. Chen, T. Farley, and N. Ye, “QoS requirements of network applications on the Internet,” *Inf.-Knowl.-Syst. Manage.*, vol. 4, no. 1, pp. 55–76, Jan. 2004.
[21] F. Delli Priscoli, M. Iannone, A. Pietrabissa, and V. Suraci, “Modelling quality of experience in future Internet networks,” in *Proc. Future Netw. Mobile Summit*, Berlin, Germany, Jul. 2012, pp. 1–9.
[22] M. Volk, J. Sterle, U. Sedlar, and A. Kos, “An approach to modeling and control of QoE in next generation networks,” *IEEE Commun. Mag.*, vol. 48, no. 8, pp. 126–135, Aug. 2010.
[23] Y. Gong, F. Yang, L. Huang, and S. Su, “Model-based approach to measuring quality of experience,” in *Proc. 1st Int. Conf. Emerging Netw. Intell.*, Oct. 11–16, 2009, pp. 29–32.
[24] H. J. Kim *et al.*, “The QoE evaluation method through the QoS–QoE correlation model,” in *Proc. IEEE 4th Int. Conf. Netw. Comput. Adv. Inf. Manage.*, 2008, pp. 719–725.
[25] K. U. R. Laghari and K. Connelly, “Toward total quality of experience: A QoE model in a communication ecosystem,” *IEEE Commun. Mag.*, vol. 50, no. 4, pp. 58–65, Apr. 2012.
[26] A. Palo *et al.*, “A common open interface to programmatically control and supervise open networks in the future Internet,” in *Proc. Future Netw. Mobile Summit*, Lisbon, Portugal, Jul. 2013, pp. 1–9.
[27] M. S. Mushtaq, B. Augustin, and A. Mellouk, “Empirical study based on machine learning approach to assess the QoS/QoE correlation,” in *Proc. 17th Eur. Conf. NOC*, 2012, pp. 1–7.
[28] OPNET Technologies, Inc., OPNET Modeler 16.5A, Bethesda, MD, USA. [Online]. Available: www.opnet.com

C. Bruni was born in Rome, Italy, on March 3, 1939. He received the Bachelor’s degree in electrical engineering and the “Libera Docenza” degree in automatic control from the University of Rome “La Sapienza,” Rome, in 1963 and 1971, respectively.

He was an Assistant Professor in 1966, an Associate Professor in 1969, and a Full Professor in 1974 with the University of Ancona. He is currently a Full Professor of optimal control with the University of Rome “La Sapienza,” where he is the Director of the Ph.D. course in systems engineering, the Director of the postlauream courses in “Methods for Analysis, Control and Optimization of Systems” and “Methods for Biomedical Signals and Image Processing,” and the Director of the Research Center for Biomedical Systems. He has authored numerous publications in the area of system estimation, identification, and optimization.



F. Delli Priscoli was born in Rome, Italy, in 1962. He received the Bachelor’s degree (*summa cum laude*) in electronic engineering and the Ph.D. degree in systems engineering from the University of Rome “La Sapienza,” Rome, in 1986 and 1991, respectively.

From 1986 to 1991, he was with the “Studies and Experimentation” Department, Telespazio, Rome. Since 1991, he has been with the University of Rome “La Sapienza,” where he is currently a Full Professor and holds the courses “automatic controls” and “control of communication and energy networks.” In the framework of his activity, he has mainly researched on resource/service/content management procedures and on cognitive techniques for telecommunication and energy networks, by largely adopting control-based methodologies. He was/is the scientific responsible, at the University of Rome “La Sapienza,” for 31 projects financed by the European Union (Fourth, Fifth, Sixth, and Seventh Framework Programmes) or by the European Space Agency, as well as for many national projects and cooperations with major industries. He has authored about 180 papers, which appeared on major international reviews (about 65), about ten books, and about 120 conferences. He is the holder of five patents. His present research interests concern closed-loop multiagent learning techniques for quality of experience (QoE) evaluation and QoE assurance in advanced communication and energy networks, as well as all related networking algorithms.

Prof. Priscoli is a member of the International Federation of Automatic Control Technical Committee on “Networked Systems.” He is an Associate Editor of Control Engineering Practice.

G. Koch was born in Rome, Italy, on October 18, 1942. He received the Bachelor’s degree in electrical engineering and the “Libera Docenza” degree in automatic control from the University of Rome “La Sapienza,” Rome, in 1966 and 1971, respectively.

In 1971, he was an Assistant Professor with the University of Rome “La Sapienza,” where he became an Associate Professor in 1972 and has been a Full Professor since 1976. He served as a Full Professor with the University of Lecce and Roma Tre University. He has authored numerous publications in the area of system estimation, identification, and optimization. His main research interests are in probability theory, mathematical statistics, linear and nonlinear stochastic dynamic systems, martingale theory, statistic filtering, stochastic model identification in biology applications, and control of communication networks.

A. Palo received the Master’s degree in telecommunications engineering and the Ph.D. degree in systems engineering from the University of Rome “La Sapienza,” Rome, in 2009 and 2013, respectively.

Since 2010, he has been a Researcher with the Consortium for Research in Automation and Telecommunication. Since 2010, he has been working in the OMEGA and FI-WARE European projects, where he developed load balancing protocols, network-aware applications, software-defined networks interfaces, and network virtualization solutions. He has authored over ten papers on these topics.

A. Pietrabissa received the Master's degree in electronic engineering and the Ph.D. degree in systems engineering from the University of Rome "La Sapienza," Rome, in 2000 and 2004, respectively.

Since 2010, he has been an Assistant Professor with the Department of Computer, System and Management Engineering, University of Rome "La Sapienza." Since 2007, he has been a member of the Technical Committee of the Consortium for Research in Automation and Telecommunication. Since 2000, he has been participating in 12 European Union projects, two European Space Agency projects, and three national projects on telecommunications. His research focus is the application of system and control theory methodologies to telecommunication networks, with specific interest to the design of resource management protocols (e.g., connection admission control, congestion control, routing, and medium access control) for multimedia broadband satellite systems, wireless networks, and next-generation heterogeneous networks. He has authored over 20 journal papers and over 60 conference papers on these topics.