An Automated Policy-Based Management Framework for Differentiated Communication Systems

Nancy Samaan, *Student Member, IEEE,* and Ahmed Karmouch, *Member, IEEE*

***Abstract*—This paper presents a novel paradigm to approach the issue of autonomous policy-based management of wired/wireless differentiated communication systems. In contrast to existing management approaches which require static *a priori* policy configurations, policies are created dynamically. The proposed framework addresses the management issue from a new perspective through posing it as a problem of learning from current system behavior, while creating new policies at runtime in response to changing requirements. A hierarchical policy model is used to capture users and administrators' higher level goals into network level objectives. Given sets of network objectives and constraints, policies are assembled at runtime. The new approach gives more flexibility to users and applications to dynamically change their quality-of-service (QoS) requirements while maintaining a smooth delivery of QoS through network monitors feedback. Simulation results demonstrate the performance of the proposed work.**

***Index Terms*—Adaptive policies, autonomic communication systems, policy-based management, quality-of-service (QoS).**

I. INTRODUCTION

THE RELENTLESS growth in wired/wireless communication technologies coupled with rapid advances in real-time applications renders the management of such systems a major challenge. As these systems continue to grow in size, sophistication and geographic reach, application developers are introducing new nontraditional real-time applications such as video-on-demand, virtual collaborative environments, and interactive gaming with stringent time and network resources constraints. A direct consequence of these advances is the continuous struggle of management system paradigms to keep pace with the continuously increasing demand of optimal and adaptive configuration and supervision of the underlying systems. Furthermore, the continuous drive in communications technologies toward the notion of pervasive computing stipulates a profound shift in approaching the issue of communication system management and introduces challenges that could not have been foreseen in the past few years.

One aspect of management that best exemplifies the magnitude of management challenges is typically encountered in quality-of-service (QoS) provisioning. QoS management refers to the activities of QoS specification, negotiation, monitoring, and control of network resources to meet end-to-end users and applications requirements, business objectives and resource availability. The notion of predefined policies has been introduced as a promising solution to address the needs of QoS traffic management. These policies prescribe a set of rules that guide the behavior of network components. Once defined by administrators or network operators, these policies are translated into network-level policies and stored in a policy repository where it is subsequently retrieved and enforced as needed. Although significant research work has been carried out in the area of policy-based management, e.g., [1]–[4], existing techniques mainly focus on defining *a priori* policies configurations to manage network devices.

If strictly approached in that sense, policies would lend themselves to be of a static nature and thereby introduce an immediate burden on network administrators. Administrators must be able to develop different policies in order to identify and prioritize users/applications and their requirements, provision network resources, as well as monitor performance within the network. With the increasing magnitude and complexity of current system components, this task places proportional demands on administrators.

Another direct consequence of that static nature is that network administrators recourse to using estimates of network traffic and users requirements in configuring network policies. These estimates can be a major source of inflexibility. For example, initial distribution of network resources to different classes of network services may not come close to optimal utilization of these resources without factoring in past and future forecasted traffic loads. On the other hand, since network-level policies are derived from business objectives and users/applications requirements described in service level agreements (SLAs) [5], policy-based management tools have to evolve and adapt with changes in these objectives and requirements in a timely manner. For example, currently developed adaptive applications, such as multimedia applications, are designed to continuously adapt their QoS demands at runtime; thus putting a stringent dynamically changing requirement on communication systems management tools. Also, users should be involved in a QoS selection as they roam across different domains where cost is of primary concern. Another example of the limitations of current policy-based management systems is encountered during the realization of new services. With each newly installed service, the administrator will have to explicitly specify different QoS parameters such as delay,

---

Manuscript received October 1, 2004; revised April 30, 2005. This work was supported in part by the National Science and Engineering Research Council of Canada (NSERC).

The authors are with the Multimedia and Mobile Agents Research Laboratory, School of Information Technology and Engineering (SITE), University of Ottawa, Ottawa, ON K1N-6N5, Canada (e-mail: nsamaan@site.uottawa.ca; karmouch@site.uottawa.ca).

Digital Object Identifier 10.1109/JSAC.2005.857191

throughput, error rates, and availability. This specification of the service is essentially static in that it often assumes a single type of service is provided at all times, regardless of the current underlying network status. Furthermore, the configuration of an alternative adaptation strategy for this service in case of a network congestion is also static in nature.

This paper proposes a new paradigm to approach the issue of autonomous policy-based management of wired/wireless communication systems. The proposed work is inspired from the process of public policymaking process in the real-world [6]. The novelty of the presented framework lies in that sets of policies, that are specifically adapted to suit current resources availability and users' demands, are dynamically assembled and dispatched at runtime. By decoupling the functionality of adapting network-level policies from the task of mapping business objectives and abstract users requirements, the proposed work offers users and administrators the freedom to specify and dynamically change their requirements. A hierarchical policy model is used to facilitate the mapping of higher level abstract users/applications policies into network-level objectives. Given sets of network-level constraints, objectives, and sets of possible actions to be taken, decisions for policy customizations are taken at runtime based on values obtained from forecast functions to best utilize the available network resources.

The remainder of this paper is organized as follows. In Section II, related work and existing approaches for QoS management and policy adaptation are briefly discussed. Section III presents an overview of the proposed framework for autonomous management of communication systems. Sections IV and V discuss details of the adopted policy hierarchy and the runtime policy adaptation process, respectively. Simulation results and performance evaluation are discussed in Section VI. Finally, Section VII concludes this paper.

II. RELATED WORK AND MOTIVATION

Existing frameworks that have been developed to support QoS management mainly fall into one of two categories [7]; reservation-based and adaptation-based systems. Although adaptation seems to provide a more promising solution for network management, existing adaptation techniques still have certain limitations. These techniques usually lack an essential degree of flexibility to build upon past experiences gained from the impact of previously pursued adaptation strategies on system behavior.

Policy-based network management has been introduced as a promising solution to the problem of managing QoS-enabled networks. However, static policy configurations built *a priori* into network devices lack the flexibility and may not be sufficient to handle different changes in these underlying environments. Various research trends, e.g., [3], have highlighted the notion of policy adaptation and the central role that it can play in QoS management in policy-enabled networks. This notion of policy adaptation is becoming even more crucial as the managed systems become more complicated. However, policy adaptation is performed only through adapting policy parameters dynamically according to the current network behavior. In [10], a genetic algorithm based architecture for QoS control in an active service network has been introduced. Our work follows the same concept of adaptation based on learning while giving more flexibility through the use of policies at different layers to allow users/applications to dynamically specify their requirements in terms of high level policies.

Agents are used in [11] to represent *active policies*. The proposed architecture has a hyper-knowledge space, which is a loosely connected set of different agent groups which function as a pluggable or dynamically expandable part of the hyper-knowledge space. Active policies, which are agents themselves, can communicate with agents in the hyper-knowledge space to implement policies and retrieve information from agents. The architecture takes advantage of intelligent agents features such as the runtime negotiation of QoS requirements. However, an active policy by itself has to be created by the administrator, and once deployed to the network it remains static through its life-cycle.

III. PROPOSED FRAMEWORK

This section presents a brief overview of the proposed policy adaptation framework. Further details on the fundamental functionalities will be presented in the following sections.

We assume that the underlying communication system consists of a set of hybrid wired/wireless domains which all support *differentiated services* (DiffServ) [15]. For example, the underlying environment can be a DiffServ/Internet protocol (IP)-enabled large enterprise network comprised of several local area networks (LANs) and wireless LANs interconnected with a wide area network (WAN) through one or more access routers. Furthermore, we assume that different domains support three classes of services, expedited forwarding (EF) [16], assured forwarding (AF) [17], and best effort (BE). In DiffServ domains, complex flow-level functions are applied to edge routers that map incoming packets to the appropriate classes of service and marking them with the appropriate DiffServ code-point (DSCP), while core routers implement per-hop-behavior (PHB) functionalities that deal with flow aggregates rather than individual flows.

Edge routers are configured through different policies which specify the way incoming traffic is classified, policed, shaped and marked. On the other hand, the functionalities of core routers are specified through a set of policies that control various scheduling, queues, and buffer management configurations.

Fig. 1 presents a schematic description of the main components of the proposed framework. The central component in this framework is the automated policy adaptor (APA). The key feature in the APA design is the decoupling between the task of mapping abstract higher level goals into network-level objectives from the functionality of adapting network components behavior.

The first task is achieved through a hierarchical policy model. At the top level of this hierarchy, network administrators specify business objectives in the form of policies through a graphical user interface attached to the APA. In the same manner, users/applications are allowed to specify their requirements in a form of policies that are related to different parameters such

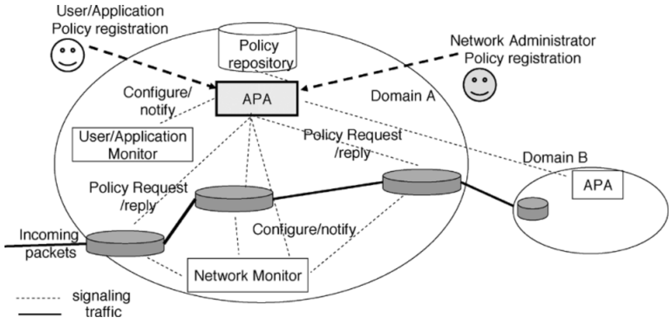


Fig. 1. Proposed policy-based management framework.

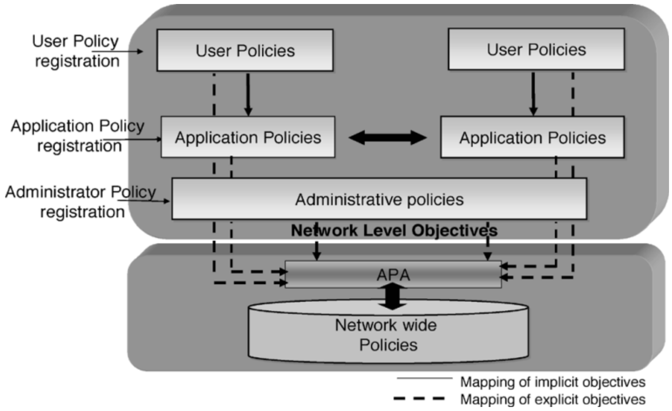


Fig. 2. Proposed APA policy hierarchy model.

as the cost of service, location of users, and time of service. The end result of this process is a set of network-level objectives. Both users and administrators register their policies with the APA.

The latter functionality of policy adaptation according to the mapped objectives is realized through an automated process. The APA interacts with edge and core routers to adapt existing policies or enforce existing ones. The APA is responsible for creating and adapting three types of policies; admission control policies, traffic conditioning policies, and provisioning policies. Admission control decisions are made at the edges of the network based on *admission control policies*. Therefore, the corresponding ingress and egress points of a requesting flow are first identified and the user's requested resources are checked to ensure that the new flow can be accommodated. Upon a successful request, the APA assembles new *traffic conditioning policies* to configure the edge routers appropriately to accommodate the new flow. The core network is also provisioned by the APA through *provisioning policies* in order to ensure that once the admission control at the edges succeeds, no bottleneck will be created in the core network. Moreover, specific policing actions are deployed to ensure that nonconforming data flows do not affect the QoS requirements for active data flows.

The adaptation process is either triggered periodically in order to slightly adjust existing policies for better performance, or triggered through events received from network monitors. In addition, adaptation is carried out according to users preregistered policies in response to events related to users and running applications. Examples of events are low battery level, movements of users, or change in an application QoS requirements. The following two sections explain the idea of the policy hierarchy model and describe the steps of the policy adaptation process, respectively.

IV. PROPOSED POLICY HIERARCHY

This section describes the methodology of translating abstract higher level users' preferences and business goals for QoS into network-level objectives. This task is not trivial due to the dynamics of these requirements and the complexity of the underlying wired/wireless environment. To illustrate those challenges, consider a scenario of a mobile user running a multimedia application. The quality of the played video stream exerts an increasing demand on the battery power of the user's terminal. Naturally, at a certain point, the user would prefer to reduce the quality of the played video if the power level drops below a certain level in order to prolong the operations of his terminal. With that consideration in mind, the user's preference for the video quality becomes a function of the battery level, which in turn is a function of time. Furthermore, assume that the cost of the delivered video service varies as the user moves between different network domains, and that the user is interested in obtaining the cheapest service as he keeps roaming across those domains. Under these circumstances, the user would have to frequently intervene manually to readjust his QoS specification in different domains and be aware of the underlying network domains and their associated operating costs. On the other hand, the running adaptive video application may select different actions in order to reduce the video quality such as: 1) reduce the frame rate by dropping frames; 2) reduce the image quality per frame; or 3) reduce the frame size. The selection of the appropriate action will depend on the user's preferences at a particular instant. This further burdens the user with having to be involved in the selection of the quality reduction mechanism.

It should be noted that this inconvenience is only attributed to having the user define his preferences in a statical manner. In the proposed work, the key solution to circumvent these problems is achieved through incorporating a hierarchical approach that enables users/applications to specify their QoS preferences in terms of policy structures. Fig. 2 depicts a schematic illustration for the proposed hierarchy. The hierarchy consists of four layers. The main concern of this section is the first three layers, i.e., users, applications, and administrative policies. Each layer i, $i = 1$, 2, and 3, is defined on a space $\bar{\mathbb{S}}_i : \bar{\mathbb{C}}_i \times \bar{\mathbb{A}}_i$. $\bar{\mathbb{C}}_i$ and $\bar{\mathbb{A}}_i$ represent the spaces of possible conditions and actions of layer i, respectively. Table I presents a sample of possible conditions and actions for each layer. In the first and third layers, users and administrators, respectively, specify their policies through a graphical user interface attached to the APA. In the second layer, applications interact directly with the APA to register their policies. Policies conditions are translated into events that the APA registers with the user/application and network monitors. Conversely, actions are translated into network-level objectives.

The task of translating parameters of QoS at each layer to network objectives, therefore, reduces to the mapping from layer i to layer j. An illustration of possible mapping methodologies can be found in [18]–[20]. As shown in Fig. 2, the two shaded

TABLE I
SAMPLE OF POLICY CONDITIONS AND ACTIONS AT DIFFERENT LEVELS OF THE APA POLICY HIERARCHY

Layer	Conditions	Actions
User policies	location $\neq$ home, terminal battery level $<$ 50%, video conference destination user name	low/medium video quality, low cost, availability, timeliness, delay guarantees, Information loss
Application policies	location $\neq$ domain name, $12:00 < time < 18:00$, user location	minimum/maxmum frame rate, frame size, quantization factor, color depth, frame latency
Administrative Policies	traffic source/destination domain, users addresses, class of service, time of service request	service access control, service class relative guarantees(Ex: bandwidth allocated to gold class $>$ 40% all other traffic)

boxes are used to show the separation between the functionalities of mapping users, applications, and administrators' policies into network-level objectives, as marked by the upper level shaded box, and the task of adapting network-level policies, as marked by the lower level shaded box. As indicated by the figure, the mapping process can be performed either between two consecutive layers or between network-level objectives and any of the other three layers. While the first mapping is referred to as an explicit mapping, the second is referred to as an implicit mapping. An example of an explicit mapping is mapping of a user policy $if(location = home)\ then\ service\ cost = 30\ \mathrm{c/min}$ into a network-level objective concerning service cost if the user's current cell ID belongs to his home network. On the other hand, an example of an implicit mapping is the mapping of a user request of a medium quality video stream into a specific frame rate and size, first, then mapping the resulting values into the required network bandwidth and delay. In general, the purpose of the differentiation between these two types is to further facilitate the automation of the mapping process. Fig. 4 depicts an illustrative example of mapping a user level policy into network-level objectives.

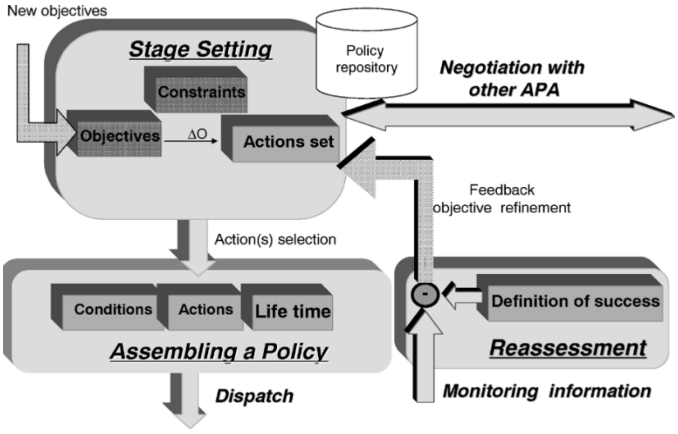


Fig. 3. Stages of adaptive policies assembly and modification.

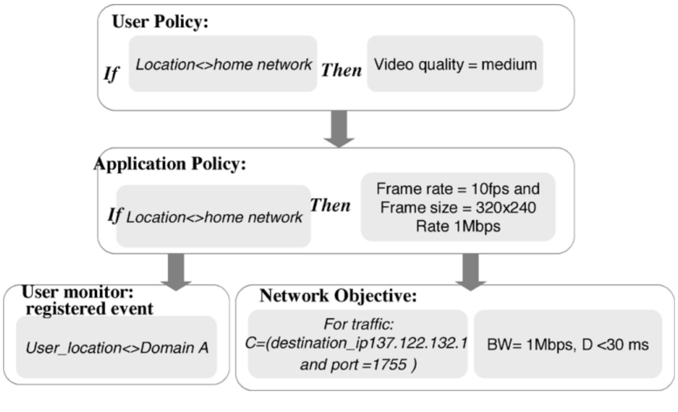


Fig. 4. Illustrative example of a policy mapping.

V. PROPOSED POLICY ADAPTATION APPROACH

In this section, we discuss the adaptation of network policies either to satisfy new users/applications and business goals or in response to feedback information reported back by network monitoring components. Earlier work that addressed the issue of policy adaptation can be classified under one of two categories [3]. Schemes within the first category perform adaptation by dynamically changing different parameters of a QoS policy to specify new attributes values. In the second category, adaptation is carried out by enabling/disabling a policy from a set of predefined QoS policies at runtime. The first category is very specialized and may not cover all situations specially in highly dynamic environments. Meanwhile, policy adaptation through enabling/disabling policies does not scale well with the dynamic users' requirements and the ever changing conditions of underlying wired/wireless environments. The APA scheme tackles this issue via posing it as a problem of learning from current system behavior and using the results of this learning process to assemble new policies at runtime. Nevertheless, the main challenge in assembling policies at runtime lies in deciding on the appropriate policy actions that can be applied to the different network components. More precisely, the following two issues arise in the decision making process.

The first issue is related to the basic steps taken to reach the optimum decision. This issue is addressed and inspired by novel approaches to the process of public policy making in the real-world [6]. In these approaches, the policymaking process passes through three main phases; stage setting, consideration of alternative decisions, and finally, reassessment of the applied decision. As shown in Fig. 3, the proposed scheme follows the same steps to decide network policies, while a feedback mechanism is used to ensure the correctness of the delivered policies.

The following sections present details of these three phases and their implementation in the context of network policies.

The second key issue is in fact related to the second step of *consideration of alternative decisions* which we elaborate further in Section V-B. Of particular importance in this step is the notion of estimating the implications of each of the candidate decisions on the network performance. For that purpose, a formal prediction process that can measure the impact of each potential decision on different network components is needed. Solving this issue stems from two main observations. The first observation is that the behavior of the network in response to a certain action is typically a function of the network traffic. The second observation is that network traffic has self-similar properties and shows long-range dependencies. This observation, reported in [21], suggests that forecasting methods of economical and technical phenomena can be employed to predict the behavior of each candidate action.

Forecasting methods can roughly be classified into one of two categories: time series and causal regression [22]. Causal regression forecasts deal with the relationship between causes and consequences for long term forecasts. In time series methods, the forecast is based on the earlier behavior. Since time series methods proved to be successful in estimating network traffic behavior, we adopt the same methods for forecasting the behavior of different actions [23]. Specific details on the adopted forecasting methodology will be presented in the following sections.

A. Stage Setting: Information Gathering

The first step in the adaptive policy making process is *stage setting*. At this stage, all information necessary for the process of policy creation is obtained. The information collected at this stage is defined over three spaces: *constraints space* ($\mathbb{CN}$), *objectives space* ($\mathbb{O}$), and the space of all possible *actions* ($\mathbb{A}$).

The constraints space $\mathbb{CN}$ represents the space of all physical limitations that can be imposed by the routers features. Examples of these constraints are: memory size, CPU speed, link bandwidth, and available mechanisms for policy implementations (e.g., available classifier types, metering, shaping, and policing mechanisms, and scheduling and queueing disciplines).

The objectives space $\mathbb{O}$ of the APA describes the overall network QoS guarantees for individual flows, as well as for different classes of services. Formally, $\mathbb{O}$ is defined as the cartesian product $\mathbb{O} : \mathbb{C} \times \mathbb{BW} \times \mathbb{L} \times \mathbb{D} \times \mathbb{J}$, where

- $\mathbb{C}$ is the space of classifications that a traffic may belong to. Therefore, an element $C_i \in \mathbb{C}$ can be a class of service for flow aggregates, e.g., EF and AF, or an individual flow specification, e.g., IP source/destination addresses, port numbers, MAC addresses, or combination thereof.
- $\mathbb{BW}, \mathbb{L}, \mathbb{D}$, and $\mathbb{J}$ are the spaces of allowed values of bandwidth, delay, jitter and loss, respectively.

An objective $O_i(C_i, BW_i, L_i, D_i, J_i) \in \mathbb{O}$, or O_i for short, describes the bandwidth, loss, delay, and jitter requirements for a traffic within the classification C_i. Examples of two APA objectives are: $O_1(\mathrm{EF}, 30\%, 1\%, 20\text{ ms}, 3\text{ ms})$ and $O_2(\mathrm{IP\_source} = 122.145.1.20, 5\text{ Mb/s}, 3\%, 40\text{ ms}, 5\text{ ms})$.

The set of network objectives, at a certain time t, $O_t \subset \mathbb{C}$, can then be expressed as $O_t = \{O_0^t, O_1^t, O_2^t, \cdots, O_n^t\}, O_i^t \in \mathbb{O}$. O_{t_0} is a statically configured objective which represents the goal of maximizing the utilization of the available bandwidth capacity. It is worth mentioning here that objectives may take absolute values, e.g., delay guarantees of EF class is less than 100 ms, $D_{\mathrm{EF}} < 100$ ms, or relative to other objectives values, e.g., delay guarantees of EF class is less one third of the delay guarantees of AF class $D_{\mathrm{EF}} < 1/3 D_{\mathrm{AF}}$. An objective $O_i^t \in O_t$ will change in response to changes in higher level policies.

Now, as pointed out in the previous section, APA policy adaptation is triggered due to either a translated change in user/application or business policies or due to a feedback from network monitors. Hence, once the adaptation process is triggered, the input to the stage setting step can be represented as a partial subset $\hat{O} = \{\hat{O}_1 \hat{O}_2, \cdots, \hat{O}_n\}$ reflecting the required new objectives or the actual measured objective, respectively. As the new subobjectives are fed to this stage, an *objectives change* $\Delta O = \{\Delta O_1, \cdots, \Delta O_m\}$ is calculated as follows:

$$\begin{aligned}\Delta O &= \hat{O} - O_t \\ &= \Big\{\Delta O_i \| \forall \hat{O}_i \in \hat{O} : \Delta O_i(C_i, \Delta BW_i, \Delta L_i, \Delta D_i, \Delta J_i) \\ &\qquad = \hat{O}_i(\hat{C}_i, B\hat{W}_i, \hat{L}_i, \hat{D}_i, \hat{J}_i) \\ &\qquad\quad - O_j\left(C_j^t, BW_j^t, L_j^t, D_j^t, J_j^t\right), C_j^t = \hat{C}_i\Big\}. \qquad (1)\end{aligned}$$

The objective change ΔO is then used in the next stage of candidate action selection.

Finally, the actions space $\mathbb{A}$ includes possible actions that can be used in the action part of the policies. Associated with each action is an indicator of whether this action can be applied to an edge or a core router, i.e., this indicator specifies whether this action is applied to a single flow or flow aggregates.

Each action is associated with a *forecast function (FF)* [22]. *FF* is a mapping of a certain action A_k to a set of forecasted values. This set of forecasted values is represented by a sparse matrix $\mathcal{FF}_k$, written using column-wise notation as follows:

$$\mathcal{FF}_k(n) = \left[ff_1^k(n) | ff_2^k(n) | \cdots | ff_m^k(n)\right].$$

In general, $\mathcal{FF}_k(n)$ describes an estimated forecast of the effects of the nth application of action A_k on network performance. More precisely, the jth entry of the ith column, $ff_{j,i}^k(n)$, represents the forecasted change in the jth QoS parameter for a certain flow with characteristic classification C_i in response to applying network action A_k. The index j, therefore, can take values of 1, 2, 3, and 4 corresponding to bandwidth, loss, delay, and jitter, respectively.

Future values of each entry of $\mathcal{FF}_k$ at the next step $n+1$ can be obtained by using one of the established forecasting techniques [22]. In this paper, we adopt a general forecasting function with desirable smoothing features; namely, the adaptive-response-rate single exponential smoothing (ARRSES) function. Using the ARRSES function, the value of the (j, i) entry in $\mathcal{FF}_k(n+1)$ is calculated as follows:

$$\begin{aligned}ff_{j,i}^k(n+1) &= \alpha_n \partial Y_{n+1}^j + (1 - \alpha_n) ff_{j,i}^k(n) \\ \alpha_n &= \left|\frac{E_{n-1}}{M_{n-1}}\right|, \; E_{n-1} = \beta e_{n-1} + (1-\beta) E_{n-2} \\ M_{n-1} &= \beta |e_{n-1}| + (1 - \beta) M_{n-2}, \; e_{n-1} = Y_n - ff_{i,j}^k(n) \end{aligned} \qquad (2)$$

TABLE II
APA OBJECTIVES

O	C	BW	loss rate(%)	avg. delay(ms)
O_1	EF	30%	0.1	30ms
O_2	AF11	25%	0.15	100ms
..				...
O_i	BE	20%	-	-
O_{i+1}	IP_source = 122.137.120.1	1Mbps	0.4	100ms

TABLE III
SAMPLE OF APA ACTIONS FOR AN EDGE ROUTER

A	Action
A_1	Increase token bucket generation rate by 1% for flow C_i
A_2	Decrease token bucket generation rate by 1% for flow C_i
A_3	Increase token bucket burst (size) for flow f
A_4	remark with AF
A_5	Increase peak burst size of Three color meter by 1% for flow f
A_6	Add dropper
	$\vdots$
A_n	Report back to application to perform adaptation

where $0 < \beta < 1$, and $|.|$ denotes absolute values. $ff^k_{j,i}(n)$ is the past effect of action A_k on the jth QoS parameter of a traffic with characteristics C_i, calculated after n times of enforcing A_k. $ff^k_{j,i}(n+1)$ is the forecasted value of the change in the jth parameter of a traffic C_i if A_k is to be applied.

∂Y^j_{n+1} is the observed effects of A_k on the jth QoS parameter of a traffic with characteristics C_i as reported by network monitors.

It can be noticed that the value of the controlling parameter, α_n, is defined as the absolute value of the ratio of a smoothed error E_t and a smoothed absolute error M_t. In practice, an administrator can either select one *best* forecasting model for all actions, or develop a rule that will select the best model for each action behavior. Modeling a system behavior is a well-studied field of research [24], [25]. However, to maintain lucidity in illustrating the basic contribution of the proposed adaptation scheme, we considered the utilization of one simple forecasting model, namely, the ARRSES model. ARRSES has been chosen for its simplicity, low memory requirements, and accuracy. Furthermore, it adjusts itself by changing the value of α_n to follow basic changes in the action effects on different QoS parameters. This provides an automated way of following the effects of different traffic loads over the network.

TABLE IV
SAMPLE OF APA ACTIONS FOR A CORE ROUTER

A	Action
A_1	Increase weighted round robin priority for class C_i by 1%
A_2	Decrease weighted round robin priority for class C_i by 1%
	...
A_m	Increase drop tail queue capacity of class C_i by 1%

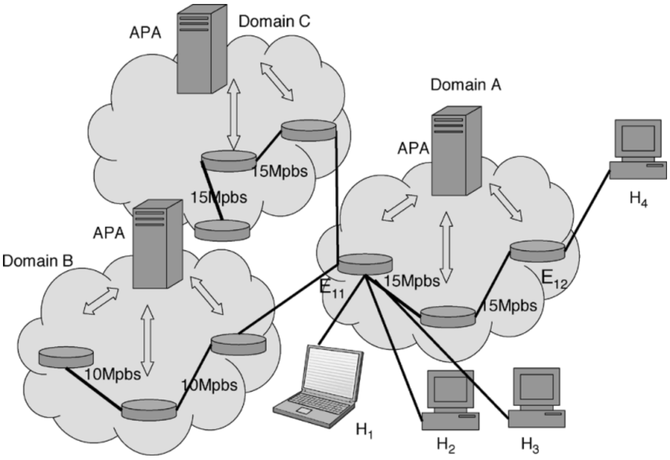


Fig. 5. Experimental testbed.

B. Candidate Actions Selection

In the second step, the APA selects one or more actions from the actions space that best attain the specified change of objectives ΔO obtained from (1).

The selection process is carried out by calculating an *expected loss value* for each action $A_k \in \mathbb{A}$, $\mathcal{E}_{\mathcal{L}}(A_k)$, as $\mathcal{E}_{\mathcal{L}}(A_k) = \sum_i \sum_{j=1}^{4} w_j(\Delta O_{j,i} - ff^k_{j,i}(n))$, $A_k \in \mathbb{A}$, where $\Delta O_{j,i}$ represents the required objective change in the jth QoS parameter of class C_i, and is obtained from (1). w_j is the weight of significance for the jth QoS parameter. In other words, $\mathcal{E}_{\mathcal{L}}(A_k)$ represents the discrepancy between the forecasted change in the QoS parameters and the actual requested change.

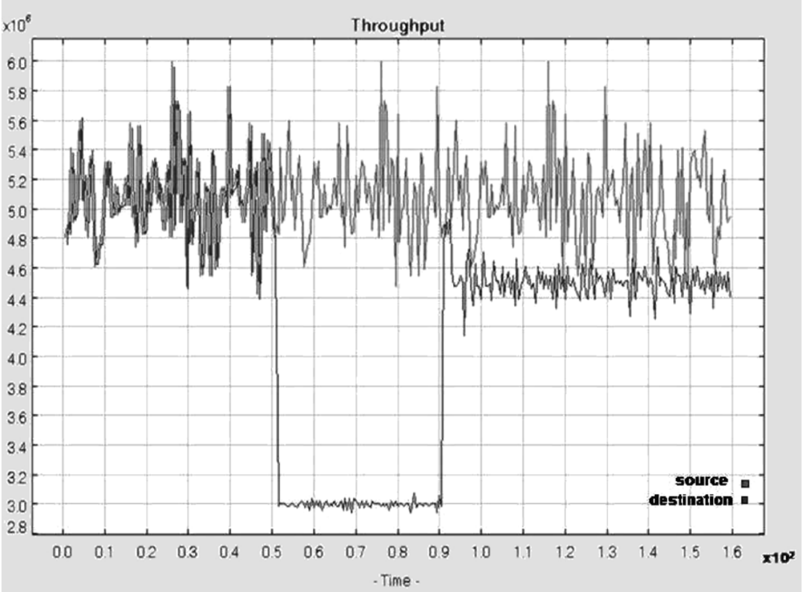


Fig. 6. Adaptation based on user policy.

Now, the process of choosing the optimum policy action reduces to the task of selecting an action $\tilde{A}$ that minimizes the expected loss values. This idea can be formally stated as follows:

$$\tilde{A} = \min_{A_k \in \mathbb{A}} \mathcal{E}_{\mathcal{L}}(A_k). \tag{3}$$

At this step, the APA may negotiate with other neighboring APAs for the selection of the most suitable actions to be taken. In general, the APA negotiates with neighboring APAs through query and response messages. This negotiation process can be performed through the utilization of any signaling protocol, e.g., common open policy service (COPS) [26]. The negotiation may also involve the exchange of different actions and values of their forecasting functions. For example, the APA located at the domain of a source node will contain the value of the forecasting function associated of an admission control policy applied to traffic originating from that specific source. On the other hand, the value of the forecasting function associated with the class of service to which this traffic belongs to at different domains is located at different APAs along the path between the source and the destination domains. In this case, each APA will separately prepare and exchange a list of different possible alternative actions and their associated forecasted values. Based on these values, the action that minimizes the expected loss values is selected.

It is also worth noting that, in addition to the negotiation for the action selection process, APAs negotiate to perform various normal management tasks. One example is testing the feasibility of an application's requirements determined through a newly requested objective. In this case, the APA at the source's domain will negotiate with other APAs along the source-destination path for the availability of resources with regard to the priority of the requesting application. In addition, APAs also communicate to exchange monitoring information.

Finally, to reduce the computational overheard required in calculating the *expected loss value* for each action A_k during the candidate action selection, actions can be further classified according to their effects on various parameters such as the bandwidth and delay. In this case, only the subset of actions that satisfy the general requirements of a certain objective, such as increasing the bandwidth or decreasing the delay will be evaluated as the candidate actions in the selection process. However, to maintain lucidity in presenting the proposed scheme, the classification of actions is only limited to indicate whether an action is considered as an edge or a core action.

C. Policy Assembly

The third step involves the assembly of a new policy given the actions selected in the previous step. The assembled policy consists of a triggering event as translated by the APA from higher level policies (e.g., user location), conditions specified by the characteristics C_i of the satisfied objective, and the selected action $\tilde{A}$. The new policy can also be associated with a life time after which it should expire and be deleted. Once a policy is assembled, it is dispatched to be applied at the network level either to an ingress, egress, or a core router according to the selected action type.

D. Reassessment

The final step in the policy adaptation process is performed by the *reassessment module*, as shown in Fig. 3, in order to evaluate the degree of success of the previously dispatched policy.

Network monitors measure the average values for the actual QoS parameters of different classes. For example, the actual throughput of traffic of class C_i at time T is calculated as follows:

$$BW_i^{\text{avg}}(T) = \frac{1}{T} \int_{t_0}^{t_0+T} BW_i(t)dt. \tag{4}$$

In a similar manner, other performance parameters, $L_i^{\text{avg}}(T)$, $D_i^{\text{avg}}(T)$, and $J_i^{\text{avg}}(T)$, are calculated for each class of service C_i.

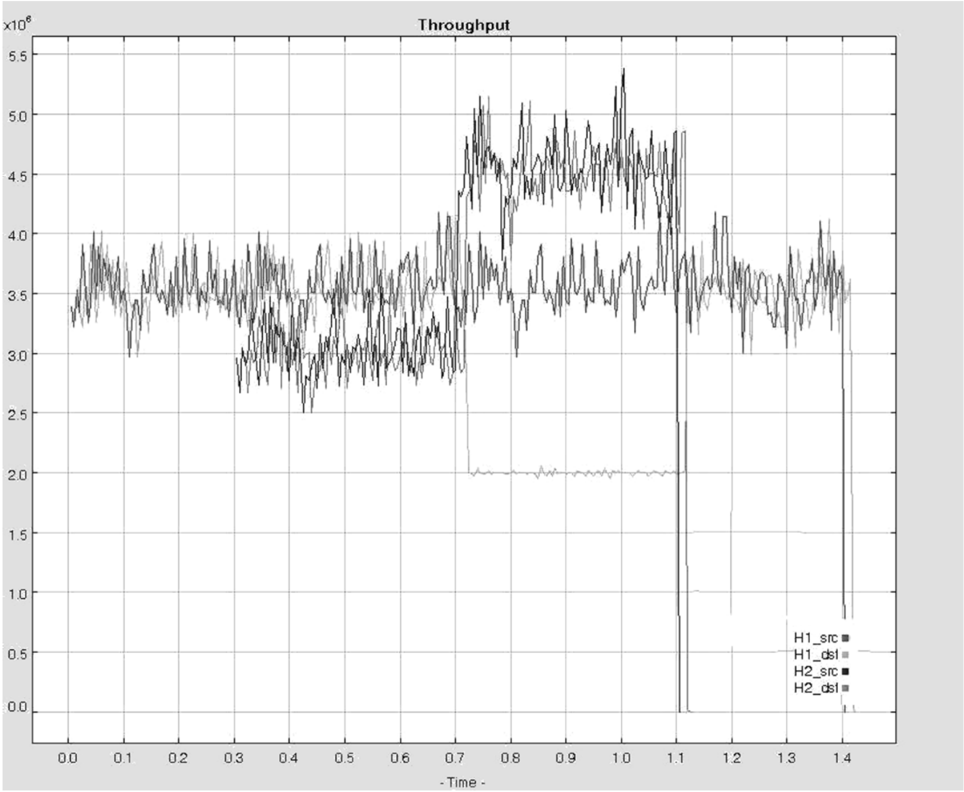


Fig. 7. Adaptation based on application policy.

The APA uses a *definition of success* ($\mathcal{SUC}$) to reassess the decision of the applied policies. $\mathcal{SUC}$ is defined as the difference between the current measured values of QoS parameters and the required objectives, as follows:

$$\mathcal{SUC} = \sum_i \sum_j w_j \left(\hat{O}_{j,i} - X_{j,i}^{\text{avg}} - \mathcal{E}_{j,i} \right), \quad \mathcal{E}_{j,i} = \log \left(\frac{X_{j,i}^{\text{avg}}}{\hat{O}_{j,i}} \right). \tag{5}$$

$\hat{O}_{j,i}$ is the new network objective for the jth QoS parameter of traffic C_i. $X_{j,i}^{\text{avg}}$ is the measured value for the jth QoS parameter as obtained in (4). $\mathcal{E}_{ji}$ is an error function that reflects the degree of tolerance in a certain objective value. We chose the error function to be the natural logarithm of the ratio of the current and desired parameter values since it allows for a wide dynamic range of differences to be measured. Based on the value of $\mathcal{SUC}$, the reassessment module takes one of two actions. In case of successfully achieving the required objectives, it modifies an entry in $\mathcal{FF}$ of the successfully applied action $\tilde{A}$ using (2) and replaces the current network objective O_t with the newly applied objectives $\hat{O}$. Otherwise, the difference between the measured values by the monitors and the required objectives is fed back to the first stage as a new objective change and the adaptation process is repeated.

E. Illustrative Examples

- **Constraints set** ($\mathbb{CN}$): A typical set of constraints is concerned with each router resources. As an example, it states available memory, disk size, buffers sizes, CPU speed, and supported different queueing disciplines, e.g., FIFO, and round robin.

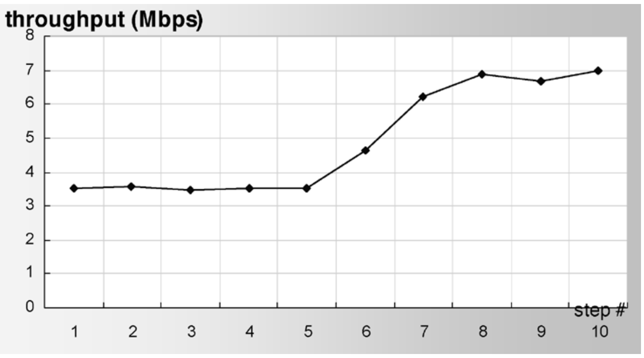
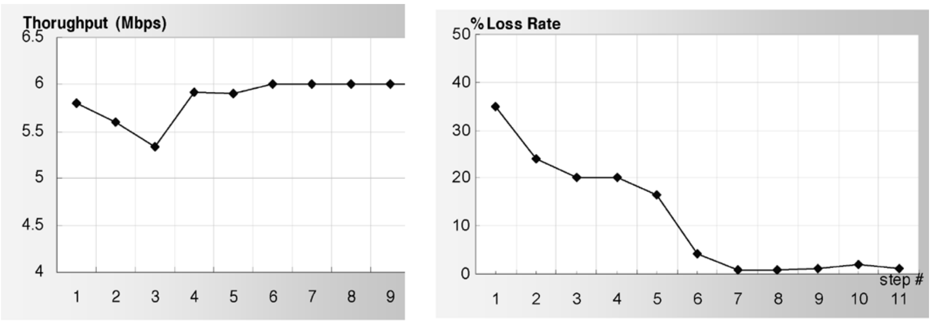


Fig. 8. Forecasted values for A_1 with respect to throughput of EF aggregate.

- **Objectives set** ($\mathbb{O}$): Table II gives an example of a possible subset of objectives of the APA. As shown, for each class of service, e.g., EF, the objective is to maintain a guaranteed level of QoS. This level is defined in terms of corresponding values for various QoS parameters such as loss rate, latency, jitter, and guaranteed bandwidth.
- **Actions space** ($\mathbb{A}$): Tables III and IV give a sample of the action spaces in edge and core routers, respectively.

VI. SIMULATION DETAILS AND RESULTS

This section summarizes simulation results of the proposed framework. A simulation architecture was constructed to model a multidomain environment used to evaluate the proposed framework. As shown in Fig. 5, each simulated domain includes an ingress router, a core router and an egress router. The simulated network has been constructed using the J-Sim



(a) EF aggregate throughput (b) EF aggregate loss rate

Fig. 9. Forecast function for A_2 with respect to EF aggregates throughput and loss rate.

network simulator [27], a simulator with a Java(tm)-based engine. The APA has been implemented on top of the simulator with a GUI to accept higher level policies that are translated into network-level objectives using heuristic rules as described in [18].

All links for domains A and C and links between all domains are configured with 15 Mb/s capacity and 5 ms propagation delay. Links in domain B are configured with 10 Mb/s with a propagation delay of 5 ms. Weighted round-robin (WRR) scheduling is implemented at each router. Initially, 40%, 30%, and 30% of the available bandwidth are set for EF, AF, and BE, respectively, for domains A, B, and C. Drop tail queues are used as buffering mechanisms for core routers. The edge router E_{11} is configured with one profile for each traffic source with initial rate as the sending rate of the source. A token bucket-based meter was used for each profile at the edge routers. Unless otherwise stated, traffic is generated by hosts H_1, H_2, and H_3 to the same destination H_4. All network traffic has been modeled with a Poisson distribution with different peak rates and a packet size of 1024 bytes. To simplify the implementation of the purposed framework, we limited our experimentation to a subset of possible actions, namely, A_1, A_2, A_3, A_4 of Table III, and A_1 and A_2 of Table IV.

A. Experiment 1: User-Based Policy Adaptation

In this experiment, we test the performance of the APA in response to a new user policy. Traffic was generated from H_4 to H_1 with sending rate of 6 Mb/s, while H_1 is moving from domain A to B at $t = 50$, and then from B to C at $t = 80$. For experimentation purposes, for EF services, we associated a cost $3c$, $5c$, and $3.3c$ per 1 Mb/s [28] for domains A, B, and C, respectively. The user on host H_1 has a registered policy P_1: $if(location \neq home\_domain)\ then\ cost < 1.5c$ *per unit time*. As the user moves from one domain to the other, P_1 is translated to different network-level objectives by the APA at domain A depending on the location of the user. Fig. 6 shows a comparison between the traffic at the source H_4 (source) and the traffic received by H_1 (destination). Initially, at time $t = 0$, P_1 has been mapped to a new objective $O(C = H_1, BW \leq 6$ Mb/s$)$ and action $A =$ "increase token rate by 1% Mb/s" is successively applied to allow the user a sending rate of 6 Mb/s. As the APA is notified with the event that the user is moving to domain B[1] P_1 is remapped to a new objective $O(C = H_1, BW \leq 3$ Mb/s$)$ and action: $A =$ "decrease token rate by 1%" is applied until the token rate of the user profile meter at E_{12} is set to 3 Mb/s. Finally, as the user moves to domain C, a new objective $O(C = H_1, BW \leq 4.5$ Mb/s$)$ is formulated and a new action $A =$ " increase token rate by 1%" is selected repeatedly to reach the target rate.

B. Experiment 2: Application-Based Policy Adaptation

In this experiment we illustrate how adaptation is performed at the network level in response to an application level policy. In this network configuration, EF traffic is generated from two hosts H_1 and H_2 to the same destination H_4. H_1 transmits at a rate of 3.5 Mb/s for the duration of the experiment. H_2 sends packets with a target EF service with a throughput requirements of 3.5 Mb/s from $t = 30$ to $t = 70$ and 4.5 Mb/s from $t = 70$ up to $t = 110$.

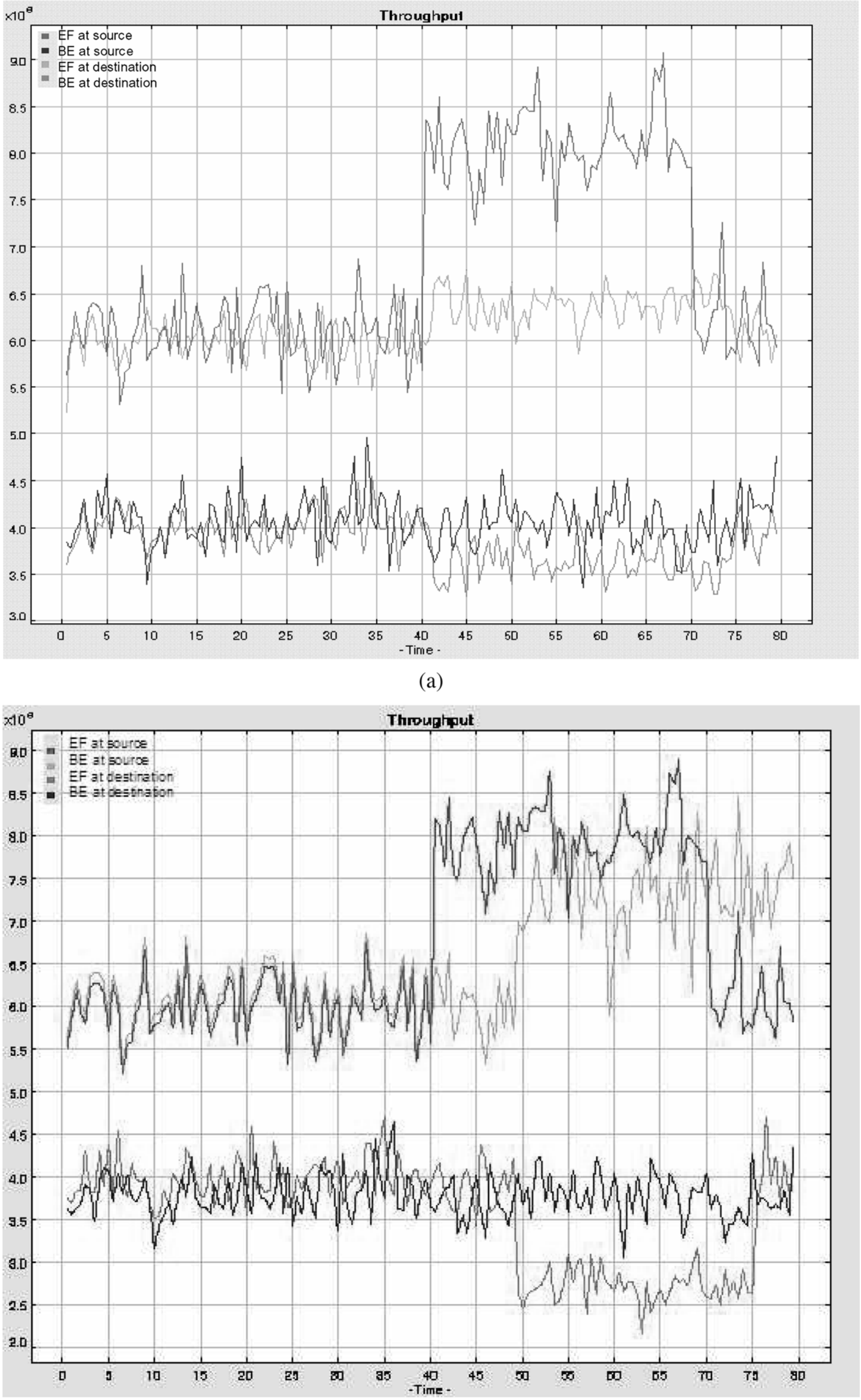
An application running on H_4 registers a policy P_2, which indicates that traffic reaching H_4 from source H_2 has double the priority of traffic from H_2, i.e., P_2: $if\ (Exists(traffic\_source = H_1\ and\ service = EF)\ and\ Exists(traffic\_source = H_2\ and\ service = EF)),\ then\ priority\ H_2\ traffic{=}\ 2\ priority\ H_1\ traffic.$

At time $t = 0$, H_1 started transmission with objective: $O_1($ source $= H_1$ and destination $= H_4$, $BW = 3.5$ Mb/s$)$. In this case, the links bandwidth is only utilized by H_1. The token bucket meter is initiated with a target rate $= 4$ Mb/s and, hence, all packets are marked with EF DSCP. At time $t = 70$, a new objective O_2(source $= H_2$ and destination $= H_4$, $BW = 3.5$ Mb/s) is fed to the APA and the old objective O_1 is replaced with a objective O_1(source $= H_1$ and destination $= D$, $BW < O_2(BW)$). The target rate of the meter of the first traffic is gradually decreased until O_1 and O_2 are satisfied.

Now, at time $t = 70$, as H_2 increases its sending rate requirements, O_2 is changed to O_2(source $= H_1$ and destination $= D$, $BW = 4.5$ Mb/s). A new action that increases the target rate for H_2 is gradually applied. Meanwhile, another action for degrading H_1 target rate is applied. Out of profile packets are remarked as AF. At time $t = 110$, the objective O_1 is replaced

---

[1] In this experiment, this event has been manually configured. However, in the real world, a user mobility prediction algorithm can be used, such as a previous work proposed by the authors [29].



(a)

(b)

Fig. 10. Throughput comparison between the statically configured network and the proposed scheme. (a) Achieved throughput in case of static configurations. (b) Achieved throughput in case of APA.

back with the initial objective and the target rate is increased to 3.5 Mb/s.

C. Experiment 3: Network Adaptation

This experiment illustrates the self-configuration feature of the proposed architecture. We illustrate how APA performs an automated selection of the appropriate actions under different network-level objectives. For simplicity, in this experiment, we limit the adaptation process to three actions: A_1, increase of WRR priority for EF by 1, A_2, increase buffer size for EF aggregates by one packet, and A_3, increase the bucket rate of single traffic of H_1 at ingress router E_{11} by 1%. All network links are set with the same capacity of 10 Mb/s. Initially, the network is configured to 60% and 40% of traffic for EF and BE, respectively. Background traffic is generated by H_2 as EF traffic with a constant bit rate of 2 Mb/s, and by H_3 as BE traffic at a constant rate of 4 Mb/s. A token bucket with rate equal to the initial sending rate of each flow is selected for each flow. H_1 transmits EF packets following a Poisson distribution with an average rate of 4 Mb/s. We start first by analyzing the ability of the selected ARRSES function in following the behavior of the applied actions. Fig. 8 depicts the forecasted values of the throughput of EF traffic aggregate in response to repetitive applications of A_1. Fig. 9(a) and (b) illustrates the forecasted values of the throughput and loss rate, respectively, of EF aggregate in response to repetitive applications of A_2.

Fig. 10 shows a comparison between the achieved bandwidth of the proposed adaptive scheme (APA) against the bandwidth achieved with static configurations of the meter, drop tail queue and WRR scheduler.

At $t = 0$, the following objectives are configured: O_1 (DSCP = EF, $BW = 6$ Mb/s, $L = 0.2\%$), O_2(source = H_1, $BW = 4$ Mb/s, $L = 0.1\%$), O_3(source = H_2, $BW = 2$ Mb/s, $L = 0.2\%$), O_4(DSCP = BE, $BW = 4$ Mb/s). Initially, the token rate of each flow is set with the values of its sending rate.

Due to feedback from the token bucket meter of the flow from H_1, an indicated loss rate >0.1% was reported to the APA. At this stage, only the objective of a single flow was not met, i.e., an objective refinement $\Delta O_2(\Delta L < 0.1)$ was triggered. At this stage action A_3 is selected for adaptation. The token bucket rate is increased gradually. This caused a slight increase in the rate of flow f_1 from traffic of H_1. However, since, at the core router, traffic from both H_1 and H_2 is being dropped by the drop tail queue after a certain period, the loss rate of EF class is increased, where O_1, O_2, and O_3 are violated.

Since action A_1 will result in violating objective O_4, A_2 is selected and the size of the drop tail queue is gradually increased. As shown in Fig. 10, this indicates a temporary increase in the throughput of both flows. However, after a certain queue size, the increase does not affect the throughput of both flows. In response to that, action A_1 is selected and repetitively applied giving priority to O_1, O_2, and O_3 over O_4.

VII. CONCLUSION AND FUTURE WORK

In this paper, we have presented a novel framework for adaptive policy-based QoS management in wired/wireless networks. The novelty of the proposed algorithm lies in that given sets of objectives, constraints and goals, network policies are assembled at runtime. It has been shown that adapting policies at runtime provides flexible means for controlling network behavior as the surrounding environment changes. In addition, it gives more freedom to users and applications providers to describe their requirements, in a continuously changing manner, in terms of policies that are functions of different parameters such as time, location, and cost. In the future, we plan to further evaluate the performance of the proposed framework through a prototype implementation.

REFERENCES

[1] P. Flegkas, P. Trimintzios, and G. Pavlou, “A policy-based quality of service management system for IP DiffServ networks,” *IEEE Netw. (Special Issue on Policy-Based Networking)*, vol. 16, no. 2, pp. 50–56, Mar./Apr. 2002.

[2] L. Lymberopoulos, E. Lupu, and M. Sloman, “QoS policy specification—a mapping from ponder to the IETF policy information model,” in *Proc. 3rd Mexican Int. Conf. Comput. Sci.*, Sep. 2001.

[3] ——, “An adaptive policy based management framework for differentiated services networks,” in *Proc. IEEE 3rd Int. Workshop Policies Distrib. Syst. Netw.*, Jun. 2002, pp. 147–158.

[4] G. Valérie, D. Sandrine, K. Brigitte, D. Gladys, and H. Eric, “Policy-based quality of service and security management for multimedia services on IP networks in the RTIPA project,” in *Proc. IFIP/IEEE Int. Conf. Manage. Multimedia Netw. Serv.*, Santa Barbara, CA, Oct. 2002, pp. 25–35.

[5] E. Bouillet, D. Mitra, and K. G. Ramakrishnan, “The structure and management of service level agreements in networks,” *IEEE J. Sel. Areas Commun.*, vol. 20, no. 4, pp. 691–699, May 2002.

[6] E. W. Walker, “Adaptive policies: an approach for dealing with structural uncertainty in public policy making,” in *Proc. 5th Int. Conf. Technol., Policy, Innovation*, The Netherlands, Jun. 2001.

[7] C. Aurrecoechea, T. Campbell, and A. and L. Hauw, “A survey of QoS architectures,” *ACM/Springer Verlag J. Multimedia Syst. (Special Issue on QoS Architecture)*, vol. 6, no. 3, pp. 138–151, May 1998.

[8] Z. Granville, L. A. Faraco de S Coelho, G. M. Almeida, and L. Tarouco, “An architecture for automated replacement of QoS policies,” in *Proc. 7th IEEE Symp. Comput. Commun.*, Italy, Jul. 2002, pp. 796–801.

[9] K. Yoshihara, M. Isomura, and H. Horiuchi, “Distributed policy-based management enabling policy adaptation on monitoring using active network technology,” in *Proc. 12th IFIP/IEEE Int. Workshop Distrib. Syst.: Oper. Manage.*, Nancy, France, Oct. 2001.

[10] I. Marshall and C. Roadknight, “Provision of quality of service for active services,” *Comput. Netw.*, vol. 36, no. 1, pp. 75–85, Jun. 2001.

[11] T. Hamada, P. Czezowski, and T. Chujo, “Policy-based management for enterprise and carrier IP networking,” *FUJITSU Sci. Tech. J.*, vol. 36, no. 2, pp. 128–139, Dec. 2000.

[12] T. Ahmed, R. Boutaba, and A. Mehaoua, “A measurement-based approach for dynamic QoS adaptation in DiffServ network,” *J. Comput. Commun.*, Mar. 2005, to be published.

[13] N. Damianou, E. Dulay, and M. Sloman, “The Ponder policy specification language,” in *Proc. IEEE 2nd Int. Workshop Policies Distrib. Syst. Netw.*, Bristol, U.K., Jan. 2001, pp. 18–39.

[14] A. Bandara, E. Lupu, J. Moffet, and A. Russo, “A goal-based approach to policy refinement,” in *Proc. 5th IEEE Workshop Policies for Distrib. Syst. Netw.*, New York, Jun. 2004, pp. 229–239.

[15] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, “An architecture for differentiated services,” IETF Network Working Group, RFC 2475, Dec. 1998.

[16] B. Davie, A. Charny, F. Baker, J. C. R. Bennett, K. Benson, J. Y. Le Boudec, W. Courtney, S. Davari, V. Firoiu, and D. Stiliadis, “An expedited forwarding PHB (per-hop behavior),” IETF Network Working Group, RFC 3246, Mar. 2002.

[17] J. Heinanen, F. Baker, W. Weiss, and J. Wroclawski, “Assured forwarding PHB,” IETF Network Working Group, RFC 2597, Jun. 1999.

[18] J. Shin, J. W. Kim, and C. J. Kuo, “Quality-of-service mapping mechanism for packet video in differentiated services network,” *IEEE Trans. Multimedia*, vol. 3, no. 2, pp. 217–230, Jun. 2001.

[19] G. Ghinea, J. P. Thomas, and R. S. Fish, “Mapping quality of perception to quality of service: The case for a dynamically reconfigurable communication system,” *J. Intell. Syst.*, vol. 10, no. 5/6, pp. 607–632, 2000.

[20] J. Shin, J. W. Kim, and C. C. Kuo, "Quality-of-service mapping mechanism for packet video in differentiated services network," *IEEE Trans. Multimedia*, vol. 3, no. 1, pp. 219–231, Jun. 2001.

[21] W. E. Leland, M. S. Taqq, W. Willinger, and D. V. Wilson, "On the self similar nature of ethernet traffic," in *Proc. ACM SIGCOMM*, Sep. 1993, pp. 183–193.

[22] S. G. Makridakis, S. Wheelwright, and R. J. Hyndman, *Forecasting: Methods and Applications*, 3rd ed. New York: Wiley, 1998.

[23] N. Papagiannaki, N. Taft, Z. Zhang, and C. Diot, "Long-term forecasting of Internet backbone traffic," *Trans. Neural Netw.*, vol. 16, no. 5, pp. 1110–1124, Sep. 2005.

[24] N. Meade, "Evidence for the selection of forecasting methods," *J. Forecasting*, vol. 19, no. 6, pp. 515–535, Nov. 2000.

[25] A. R. Venkatachalam and J. E. Sohl, "An intelligent model selection and forecasting system," *J. Forecasting*, vol. 18, no. 3, pp. 167–180, May 1999.

[26] D. Durham, J. Boyle, R. Cohen, S. Herzog, R. Rajan, and A. Sastry, "The COPS (common open policy service) protocol," IETF, RFC 2748, Jan. 2000.

[27] H.-Y. Tyan and C.-J. Hou, JavaSim On-Line Manuals and Tutorials.

[28] H. Jiang and S. Jordan, "A pricing model for high-speed networks with guaranteed QoS," in *Proc. IEEE INFOCOM*, Apr. 1996, pp. 88–95.

[29] N. Samaan and A. Karmouch, "A mobility prediction scheme based on Dempster-Shafer theory of evidence," *IEEE Trans. Mobile Comput.*, vol. 6, no. 5, pp. 537–551, Nov.–Dec. 2005.

Nancy Samaan (S'03) received the B.Sc. and M.Sc. degrees from the Department of Computer Science and Automatic Control from Alexandria University, Alexandria, Egypt, in 1998 and 2001, respectively. She is currently working towards the Ph.D. degree at the School of Information Technology and Engineering, University of Ottawa, Ottawa, ON, Canada.

Her current research interests include mobile communications, quality-of-service (QoS) issues and policy-based network management.

Ms. Samaan was a recipient of the Ontario Graduate Scholarships in Science and Technology, the Ontario Graduate Scholarship, and the University of Ottawa Excellence Scholarship.

Ahmed Karmouch (M'92) received the M.S. and Ph.D. degrees in computer science from the University of Paul Sabatier, Toulouse, France, in 1976 and 1979, respectively.

From 1976 to 1983, he was a Research Engineer at the Institute National de Recherche en Informatique et en Automatique (INRIA), Paris, France. First, in distributed databases with the Sirius Project, and then with the Kayak Project, where he researched office information systems and was responsible for the Multimedia Distributed Message System Research Group. From 1984 to 1988, he was with Bull SA, Paris, France, as a Senior Manager with the Department of Advanced Studies. He was responsible for the Distributed Multimedia Document Management Group. From 1988 to 1991, he was Director of research on Multimedia Distributed Databases and Architectures at the Ottawa Medical Communications Research Group, University of Ottawa, Ottawa, ON, Canada. Since 1991, he has been a Professor of Electrical and Computer Engineering and Computer Science at the School of Information Technology and Engineering, University of Ottawa. He also holds an Industrial Research Chair from the Ottawa Carleton Research Institute and Natural Sciences and Engineering Research Council. He has been Director of the Ottawa Carleton Institute for Electrical and Computer Engineering. He is involved in several projects with the Telecommunications Research Institute of Ontario, Nortel Networks, Bell Canada, Mitel, National Research Council Canada, Centre National de Recherche Scientique, France, March Networks, CANARIE, Communications and Information Technology Ontario (Cito), and TeleLearning National Center of Excellence. He is a partner in the "sWWI: Ambient Networks," a European Sixth Framework Integrated Project. He has served in several program committees, organized several conferences, and workshops. He has published over 160 papers in the area of multimedia systems, ad hoc communications, and mobile agents for telecommunications. His current research interests are in distributed multimedia systems and communications, mobile computing, home architecture and services, context aware ad hoc communications, and mobile software agents for telecommunications.

Dr. Karmouch is a member of the Association for Computing Machinery (ACM). He served as Guest Editor for the *IEEE Communications Magazine*, *Computer Communications*, the *International Journal of Multimedia Tools and Applications* (MTAP), and others.