Efficient agent-based selection of DiffServ SLAs over MPLS networks

Thanasis G. Papaioannou[a,b], Stelios Sartzetakis[a,*] and George D. Stamoulis[a,b]

[a] Institute of Computer Science (ICS), Foundation for Research & Technology – Hellas (FORTH)
P.O. Box 1385, GR 711 10, Heraklion, Crete, Greece

[b] Department of Computer Science, University of Crete, Heraklion, Greece

ABSTRACT

The demand for QoS provisioning support over Internet grows continuously. The most scalable and less demanding solution, in terms of necessary modifications to the existing Internet infrastructure, is the Differentiated Services (DiffServ) architecture. In this approach, little care has been taken for on a per-application or a per-user basis QoS provisioning. Nevertheless, the Service Level Agreements (SLAs) need to be selected in a way that is efficient both for the users and for the network. In this paper, we develop and evaluate an approach for efficient SLA selection and employment in a DiffServ-over-MPLS network domain. A negotiation process between a user and a network provider is introduced; thus the user can choose from alternative options for allocation of resources the one that better matches his needs. We adopt a usage-based charging scheme that provides the user with the right incentives for SLA selection. For the purposes of negotiation, we develop an appropriate utility model that expresses user preferences in a simple yet informative way. An important feature of our approach is the distribution of information enabling the SLA selection. Moreover, under this distribution, each of the components involved only possesses this information for which it has an incentive to store. We describe the implementation of our approach in a real network and discuss its features and performance. We also present experimental results regarding the overall efficiency attained by means of the SLA selection process. These results advocate that our approach is incentive compatible, in the sense that individual optimization by each user (via SLA selection) may also lead to improved social welfare. Our approach is quite generic and can be combined with several policies for network management. It can also be employed as a complement to the traffic engineering procedures.

Keywords: QoS, SLA negotiation, user utility, DiffServ, incentives, MPLS

1. INTRODUCTION

The continuously growing demand for QoS provisioning support over Internet has led researchers to define various possible solutions to the problem. The most scalable and less demanding solution, in terms of modifications to the existing Internet infrastructure, is the Differentiated Services (DiffServ) architecture[1]. It is summarized in applying a forwarding behaviour (PHB) to traffic aggregates at each hop, through a codepoint assignment at every flow according to a Service Level Agreement (SLA). While this approach is attractive for its scalability properties it has some severe drawbacks, since little care has been taken for QoS provisioning on a per-application or a per-user basis. For this reason, Integrated Services (IntServ) operation over DiffServ has been proposed[5]. Under this approach, DiffServ domains are viewed as single hops in the RSVP[2] process. Within the DiffServ domains, bandwidth for the traffic aggregates is reserved either statically or using RSVP (per traffic flow or per traffic aggregate[13]) or/and Bandwidth Brokers, as in where[4], the resource allocation process for end-to-end (across domain boundaries) SLA provision is addressed.

Multi-Protocol Label Switching[16] (MPLS) supports the notion of traffic aggregates and furthermore it can offer different QoS level for each traffic aggregate. At the same time, MPLS provides some very important features that facilitate traffic engineering. Therefore, DiffServ over MPLS seems to be a very promising approach for efficient QoS provisioning in terms of both scalability and flexibility.

[*] Correspondence: S.Sartzetakis (stelios@ics.forth.gr) – Other author e-mail: {pathan, gstamoul}@ics.forth.gr

A SLA contains among others the Service Level Specification (SLS), which can be viewed as the traffic contract. While the specification of the SLS has to be as technical and detailed as possible, in order for the network provider to accomplish resource allocation as needed, it should also be made easy for the user to specify or select a predefined SLS in a simple way. Furthermore, the SLSs have to be *negotiated* between different administrative domains as well as between administrative domains and users. Only thus, users can select the SLSs they really need and the network can allocate its resources in an efficient way, according to its particular policy. In fact, this should be done in such a way that both goals are met at the same time, which would amount to incentive compatibility.

In this paper, we develop and evaluate an approach for efficient SLA selection and employment in a DiffServ within a MPLS network domain. A negotiation process between a user and a domain is introduced; thus the user can select from alternative options for resource allocation the one that better matches his needs. We assume that services are charged according to a usage-based scheme that provides the user with the right incentives. For the purposes of selection, we develop an appropriate utility model that expresses user preferences in a simple yet informative way. The communication of the user QoS preferences to the network as well as that of the results of the resource allocation process back to the user is done simply and effectively. The remainder of this paper is organized as follows. In Section 2, we describe the overall architecture proposed for negotiation and enforcement of SLSs over a DiffServ domain. In Section 3, we describe the adopted charging scheme and justify our choice. In Section 4, we present a distribution of the information that is essential for the rationality and the scalability of our approach. In Section 5, we propose a model for user utility that expresses user preferences for SLA selection. In Section 6, we describe in detail the communication among components of the architecture during the negotiation process. In Section 7, we present some experiments that indicate that using our negotiation process for SLA selection, the right incentives for the user, which are provided by the adopted charging scheme, are maintained. In Section 8, we present specific choices that for the implementation of our architecture in a realistic network domain. In Section 9, we evaluate our approach in terms of performance. Finally, in Section 10, we present some concluding remarks and possible extensions of our work.

2. THE ARCHITECTURE

We consider a multi-hop DiffServ network domain in which telecommunication services are provided in various QoS levels. This DiffServ domain is built on top of MPLS. Multiple Connectivity Providers share the control and ownership over the network resources. The overall responsibility about the end-to-end network service provision belongs to one or many Network Service Provider(s) (NSP(s)) who has contracts with certain Connectivity Provider(s). Each user subscribes himself to a NSP in order to have access to network services. Upon subscription, a user specifies his profile, which contains information about his preferences regarding services. The user profiles are stored in a directory service and downloaded to users' hosts upon login. Also, the NSP associates a User Agent (UA) to each user, which resides on the user's host. This agent negotiates the user's SLAs on his behalf, according to his profile. The representative of the NSP is a software entity to be referred to as Policy Server (PS). The PS is responsible for making the proper decisions for resource allocation that guarantee efficiency with respect to network utilization, while enforcing incentive compatibility by means of charging. The policy directions followed by the PS are stored in a well-known Policy Directory (PD), which is unique for the network domain (e.g. restrictions on bandwidth usage or a certain charge discount for a specific user or application). A software entity, the Information Directory (ID) holds information about the current network domain state (e.g. the available bandwidth or the packet loss probability per QoS class). This information is updated over time. This way the network provider is able to apply his policy rules for specific flows or users. For performance reasons, it is preferable to have multiple instances of the same PS.

The User Agent has to select the proper SLS x in order to maximize the *net benefit* of the user, i.e to solve the maximization problem:

$$\max_{x} \{u(x) - c(x)\} \tag{1}$$

where $u(.)$ is the utility function[15.] of the user (expressing his preferences and his willingness to pay per SLS) and $c(.)$ is a function that gives the expected charge for a service provision that is compliant to an SLS. The PS can compute the expected (or, in certain cases the actual) charge for a service provision. Thus, the User Agent negotiates with the nearest PS for the proper SLS, in order to solve the above optimization problem. An important feature of our architecture is that it is necessary neither for the UA to know the location of the PS prior to the beginning of negotiation, nor for the PS to know the location of the UA. Indeed, we employ the RSVP RESV message to communicate the QoS requirements (in terms of throughput and end-to-end delay) of the user's receiving application to the NSP, if QoS provision is initiated by the receiver of the traffic. On the other hand, if the sender of the traffic has specific QoS requirements, these are communicated to the NSP, using the RSVP PATH message. For simplicity, we assume that the receiver of the traffic "absorbs" the utility induced by the various services, and so that the receiver makes resource allocation requests. Using RSVP, an application requests QoS in a clear and detailed way that is appropriate for the resource allocation process. On the other hand, the user asks for this QoS in an abstract way e.g. adjusting a slide bar in the application that expresses his preferences (e.g. improve or decrease QoS) and his willingness to pay. After SLS selection the PS associates the traffic flow of this SLS with specific MPLS resources, and finally notifies the sender of the service traffic, asking it to begin service provision according to the specified SLS. Certain QoS classes are provisioned along each path (Label Switched Path (LSP)). A NSP has to ensure uniform properties for a certain QoS class over all the LSPs in which the QoS class is provided. That is, a QoS class provides the same QoS level over all LSPs. In order to achieve that, a NSP must implement certain mechanisms for balancing the traffic load among the QoS classes on the various LSPs. However, a NSP could allow different properties for the same QoS class over different LSPs and let the users (with their SLS selection) to do the load balancing. In this case, a QoS class in different paths has been allocated different resources in terms of capacity and buffer, and thus it provides a different QoS level. The operating point parameters[a] (i.e, *space* parameter s and *time* parameter t, see Ref. 9.) and other statistical characteristics are different per QoS level. We introduce here the notion of the *noncompliance risk*, which is defined as an a priori upper bound on the percentage of traffic that will not be treated in accordance to the SLS. The noncompliance risk specifies certain requirements/properties of the distributions of the QoS parameters of a SLS; e.g., an upper bound on the end-to-end delay per packet, that can only be violated for a percentage of packets less than or equal to r. Note that, although the noncompliance risk is a factor computable by the network, it is understandable by the users too. In order to give users the right incentives, different QoS levels are associated with different charges and possibly different noncompliance risks for the same traffic flow. The PS has to find the eligible QoS levels for the traffic of each new SLA and to get the corresponding charges by the Connectivity Provider(s). The UA has to make the final selection of the SLS that maximizes the user's net benefit. For simplicity, in the rest of the paper, we assume that the NSP implements certain mechanisms for balancing the traffic load among the QoS classes on the various LSPs, thus rendering the selection of path irrelevant to the net benefit of the user. Henceforth, the term QoS class essentially refers to a QoS level.

3. THE CHARGING SCHEME

The Policy Server computes the charge for each offered SLS on the basis of effective bandwidth, using the formula below (see Ref. 3.):

$$
\begin{aligned}
&c_i(x) = \bar{a}_i(x;m)\,p_i T \\
&\bar{a}_i(x;m) = \frac{1}{s_i t_i}\log\left[1 + \frac{m}{H(t_i)}\left(e^{s_i t_i H(t_i)} - 1\right)\right] \\
&H(t_i) = \min\{h, \rho + \beta/t_i\}
\end{aligned}
\qquad (2)
$$

x in the above formula is the SLS (although the *effective peak rate* $H(t)$ suffices), $m \in [0, \rho]$ is the mean rate of the traffic to be induced by the source, the index i specifies a QoS class i. T is the duration of the flow, which is either known to the application server in advance (e.g., for a video movie), or can be estimated by the User Agent on the basis of past information. p_i is price per unit of effective bandwidth for $\bar{a}_i(x;m)$, which in theory should coincide with the *shadow price* of the constraint that limits the resources for a QoS class i. s_i, t_i parameters are the operating point of a QoS class, i.e. the operating point of the congested link[b] over the path(s) over which QoS class is implemented. In fact, $\bar{a}_i(x;m)$ is an upper

[a] The space parameter s expresses the statistical multiplexing capability of a link. If the capacity of the link is very large in relation with the peak rate of the new flow, then s=0. However, s increases if the peak rate of the new flow is large relatively to the capacity of the link. The time parameter t expresses the time prior for overload to occur. The parameters (s, t) together represent a network operating point, which depends on factors such as bandwidth, buffer sizes, traffic mix, and can be estimated from traffic measurements.

[b] The charging scheme of equation (2) is based on the assumption that all flows traverse a certain link that it the most congested one (bottleneck link), and they are charged according to resource usage on this link. If this assumption is not applicable, then the charge can be computed by adding the outcome of (2) over the path traversed by each flow. Our negotiation approach can be modified to cover this case as well, which however is more complicated.

bound on the actual effective bandwidth and corresponds to that of an on/off source. The actual effective bandwidth of a source is computed according to the so called inf-sup formula, derived by means of Large Deviations techniques[3].

This charging scheme is applicable to the calculation both of the expected charge of a new flow, and of the actual charge to be paid by the user after service provision. These two values are not expected to vary too much. This difference is due to the discrepancy between the a priori mean rate calculated by an application server for a piece of content in a QoS class and the actual mean rate with which the service is provided. This is not important, because the guarantees provided by a DiffServ network, as currently defined by the IETF, can be loose and users are actually charged according to the resources they consume, i.e. their actual mean rate measured by the network during service provision. The scheme provides users with the incentive to choose the tightest possible traffic profile (h, ρ, β), for a resource usage minimizing the actual effective bandwidth and thus the actual charge to be paid by users. Also, through negotiation, users select the QoS class that maximizes their net benefit and thus it is closer to their actual needs. As explained in Ref. 3., this charging scheme is fair (i.e. reflects actual usage), if the variance of the ratio of $\bar{\alpha}(x;m)$ to the actual effective bandwidth $\alpha(x;m)$ is small, for the set of those x and m that characterize the services of interest. Also, it gives the incentives to users to select SLSs that better match their actual needs.

Consider that we have a continuous set I of QoS classes in the DiffServ network. The variable $i \in I$ has the value 0 for the lowest and the value 1 for the highest QoS class. Thus, the value i of an intermediate QoS class can be viewed as the QoS it provides relatively to the highest QoS class. In this case the price p_i per unit of effective usage for each QoS class i can be given by an explicit function $P(i)$ of i. In general, $c_i(x)$ has to be increasing when QoS improves, in order for users to have the right incentives to select the QoS class that optimizes their net benefit (i.e. attain the best trade-off between QoS and charge).

4. DISTRIBUTION OF INFORMATION

In order for the described architecture to be feasible and scaleable, and for the traffic of a new flow to be assigned a SLS using our negotiation process, information has to be available and exchanged among the components in a proper way. In this section we describe and justify a certain distribution of information that serves these goals.

A User Agent (UA), being a representative of the user, stores the utility model and has access to the corresponding user profile. Furthermore, the user's application is able to communicate to the corresponding UA any necessary information, such as the upper bound on the noncompliance risk acceptable by the user. The Policy Server (PS) has full control of the network resources, thus being aware of the available bandwidth, the operating point (s, t), the price per unit of bandwidth and the noncompliance risk of each QoS class. The PS offers the users some information on the service classes provided, by posting the values of parameters t for each QoS class; the reason why this is useful will be explained below. In order to proceed with our negotiation approach, we have to specify how the user application determines QoS requirements (which are subsequently conveyed to the resource reservation request) as well as how the PS finds the traffic parameters for the SLSs in each QoS class. These issues are discussed below.

Each application server in the network computes and posts information regarding the performance of the services it provides. Furthermore, each piece of content is transmitted under different encoding in each service class (i.e. QoS class). For example, a particular video may be MPEG-encoded, with various quality factors. Thus, the server posts the following information: the traffic parameters in terms of token bucket parameters (ρ, β), peak rate h and mean rate m, with which the traffic of each content is sent over each class, and the duration T of the service if known. Furthermore, the server has the incentive, due to competition, to compute and employ the traffic parameters that *minimize the charge* for each piece of content in each class. A server is able to accomplish this if it is aware of the way the NSP charges. As already explained in Section 3, we assume that the NSP charges proportionally to the effective bandwidth of the user's SLS. A server has to derive the traffic parameters (ρ, β, h) that minimize this charge. The peak rate h for each piece of content in each QoS class is determined by the amount of shaping the server performs for this specific content: A smaller peak rate results from a larger amount of shaping, which however influences the playout quality due to the shaping delay introduced. As explained in Ref. 3., given a peak rate h, there are various pairs of (ρ, β) for which all the traffic sent will be conforming. These pairs belong to an indifference curve such as the one shown in Figure 1. The effective bandwidth corresponding to a SLS is increasing in the effective peak $H(t)=min\{h, \rho + \beta/t\}$. A server has simply to choose the pair (ρ, β) that minimizes H. If the minimizer of H is h, then the token bucket selection does not affect the charge. Otherwise, if the minimizer of H is $\rho+\beta/t$, then the pair (ρ^*, β^*) that minimizes H is that point in which the tangent to the indifference curve has slope $-t$; see Figure 1. For simplicity, we assume that this point is always chosen, even if $h<\rho^*+\beta^*/t$. Thus, the server derives the parameters (h, ρ^*, β^*) that minimize the finding charge for a piece of content in a QoS class that is characterized by t. It is worth noting that it is not necessary for the PS to post the values of the space parameter s. Moreover, the above procedure for selecting (ρ, β) is valid regardless of whether the duration T of the flow is known or has to be estimated.

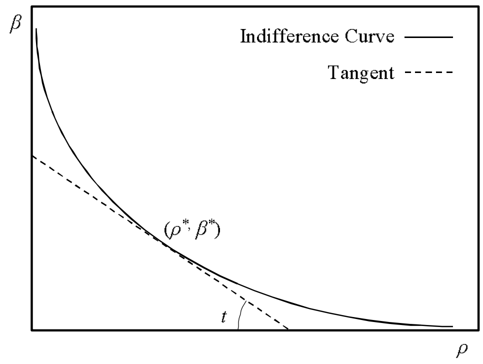


Figure 1. The indifference curve of parameters (ρ, β) for a given peak rate (shaping delay) and for a specific percentage of traffic to be conforming.

Also, a user receiving application is able to find the traffic parameters that express the user's QoS requirements. Recall that the choice of peak rate h depends on the amount of shaping delay introduced. The user application knowing the maximum acceptable shaping delay, can specify the peak rate h of the reservation request. As already mentioned, for each specific value of peak rate h there will be pairs of (ρ, β) for which all the traffic that will be received is conforming (Figure 1). The user application having information about how the NSP charges, it selects the pair (ρ^*, β^*) that minimizes charge in the required QoS class, as described in the previous paragraph. So far, we have presented a distribution of information, where each piece of information is stored by a component that has the incentive to do so. In the next sections, we explain how the components act and interact in order for the UA to make an efficient selection of the SLS.

5. MODELLING USER UTILITY

In this section we discuss how the UA selects the best SLS on behalf of the user. First, we have to model user preferences in a user utility depending on the various parameters involved in the SLS. The traffic information in this SLS includes some of the RSVP parameters, namely the peak rate h and the token bucket parameters (ρ, β). Note that a flow with the same traffic parameters (h, ρ, β) is differently served by different QoS classes, in terms of throughput, delay, jitter and packet loss. The user application asks for a specific throughput (and possibly an end-to-end delay bound) by negotiating for SLSs. Also, it specifies an upper bound for the acceptable noncompliance risk r by the network. In particular, the SLS should comprise the tuple of parameters (h, ρ, β, QoSclass, r) that maximizes the net benefit of the user. Trying to compute five parameters simultaneously is a very complicated task. Fortunately, as explained in Section 4, it can reasonably be assumed that only the best triplets (h, ρ, β) of the various QoS classes are eligible choices. Thus, it remains to select the QoS class and the noncompliance risk of a SLS.

The application servers post the traffic parameters per content in each QoS class that minimize the expected charge. In order for a user to be served by these optimized parameters per QoS class, he has to pay an amount of money per QoS class. It is assumed that the maximum amount the user is willing to pay (denoted as W_{max}) is also contained in the user profile. This amount corresponds to the best possible contract. Under the above assumptions, the UA has to select only the QoS class and the noncompliance risk values so as to maximize the user's net benefit. Assuming that the user utility does not depend on the content of the information to be received, we propose a simple utility model that serves our purposes. In particular, the utility function for an SLS x is given by the formula below:

$$u(x) = W(x) f(r_{user}, r_{netw}), \quad r_{netw} \le r_{user}$$
$$\text{where } W(x) = U(QoS_{DF}) W_{max} \tag{3}$$

In the above formula, x is the SLS in terms of peak rate h and token bucket parameters (ρ, β) *and* QoS class. $W(.)$ is a monetary expression of the user's utility for this SLS; that is, $W(.)$ is the user's willingness to pay for this traffic contract in the certain QoS class. r_{user} is the maximum percentage of noncompliance with the SLS accepted by the user and r_{netw} is the noncompliance risk expected to be offered by the network for an SLS. $f(r_{user}, r_{netw})$ is a function that expresses the user's increased satisfaction for low values of r_{netw}. The user satisfaction should be maximized (with f = 1) when r_{netw}= 0, and decrease as r_{netw} approaches r_{user}. This is due to the assumption that a user is satisfied with any r_{netw} that is lower than or equal to r_{user}. Therefore, when r_{netw} increases and approaches r_{user}, user satisfaction should decrease. In fact, we expect f to have the shape depicted in Figure 2. For each particular user the "decreasing" segment of the curve, its saddle point, and its minimum value will depend on the sensitivity of the user to the noncompliance risk offered by the network. Finally, $U(QoS_{DF})$ is the normalized utility factor that corresponds to this QoS class of DiffServ, and expresses the percentage of W_{max} that the user is willing to pay for this QoS class; this factor equals 1 for the highest such class.

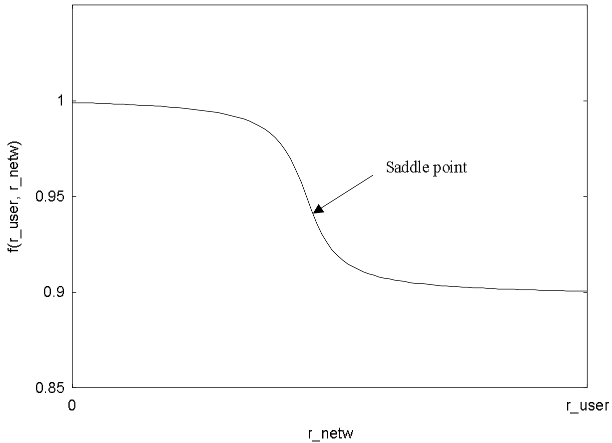


Figure 2. A possible $f(r_{user}, r_{netw})$ function.

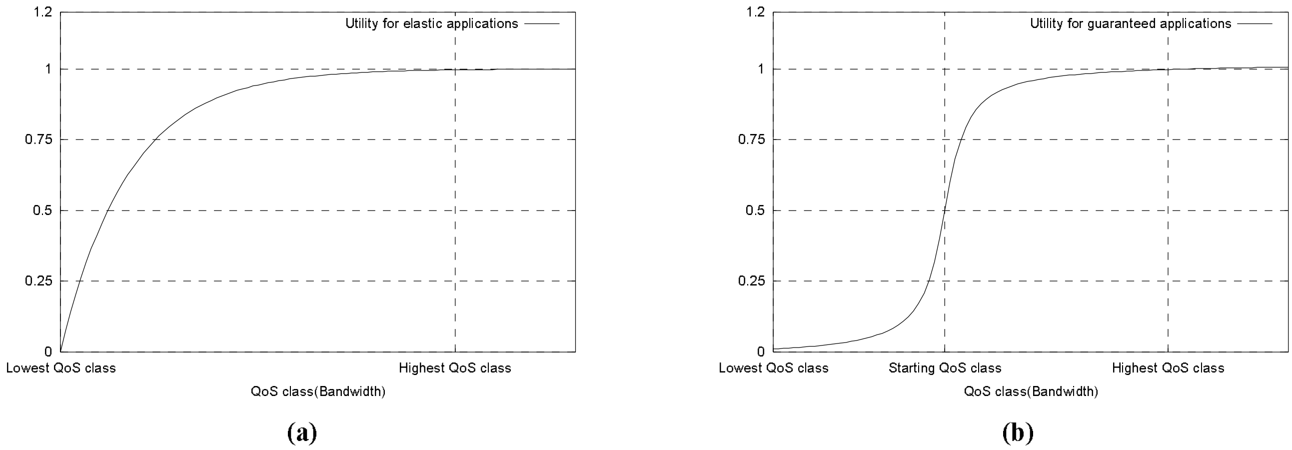


Figure 3. A normalized utility function U with respect to QoS class for elastic (a) and guaranteed (b) services.

The network services that the user asks for are either elastic or guaranteed. Note that we assume that in general a higher QoS class results in a higher scheduling priority and/or a lower dropping probability: both result in higher bandwidth consumption by the higher QoS class under the same traffic transport requirements. Thus, the shape of the function $U(.)$ (with respect to the DiffServ QoS classes) is essentially the same as that of utility expressed as a function of bandwidth, because for the same traffic transport requirements a higher QoS class consumes more bandwidth and has a lower marginal utility. For elastic services the utility function is concave, as shown in Figure 3a, due to the diminishing marginal utility induced by an additional unit of bandwidth when the amount of bandwidth already allocated increases. For guaranteed services, the utility function is initially convex and then concave with respect to bandwidth; see Figure 3b. However, as explained in Ref. 8., for guaranteed services it does not make much sense to consider the convex segment of the utility curve as acceptable, because the marginal increase in utility due to a small increase in the bandwidth to be allocated is increasing; this implies that the user has the tendency to require more and more bandwidth thus abandoning this segment. Therefore, the guaranteed utility function, as an approximation, can be taken as identical to elastic except for the fact that it starts at an index of QoS class larger than 0. The UA of a guaranteed user should know this starting point.

6. THE SLS SELECTION PROCESS

As described in Section 2, the SLS request of an application reaches the closest Policy Server (PS). The PS identifies the UA that is associated with the user of the application through the location information enclosed in the RSVP RESV message and starts negotiating on the SLS that will be finally offered. The parameters in the RSVP RESV message express the lowest bound in the QoS requested by the user. In the beginning of the negotiation process, the UA informs the PS of the parameter r_{user} that expresses the maximum acceptable value of noncompliance of the requested SLS with the SLS offered. Taking this parameter, the PS is able to communicate with the Information Directory (ID) and to find the eligible offers that satisfy the user requirements. The ID contains network state information, as the operating point (s, t), the avaivable bandwidth and the noncompliance risk of a certain QoS. Each offer is composed of QoS class, noncompliance risk r_{netw}, the parameters (s, t) of the QoS class and an expected charge $c(.)$ per time unit. In order to compute $c(.)$, the PS employs the public information about the traffic parameters for each offer in terms of peak rate h, token bucket pair (ρ, β), and the mean rate m per QoS class for the specific piece of content that is ordered from the application server. The PS communicates the offers [$c(.)$,r_{netw}, QoS class] to the UA, which solving the NetBenefit maximization problem selects the proper SLS for the user. For simplicity, we assume that this optimization is carried out exhaustively for the discrete set of QoS classes offered. The complexity of this can be improved if the concavity and monotonicity properties of the functions involved are exploited. If the maximum net benefit in the SLS selection process for a new flow is negative, i.e. the negotiation process does not find a SLS that serves the QoS requirements of a user for the flow, given a relatively low willingness to pay or a highly loaded network domain for the required QoS classes, the traffic of the new flow is serviced as Best Effort. The sequence of messages exchanged between the components during the SLS selection process are depicted in Figure 4.

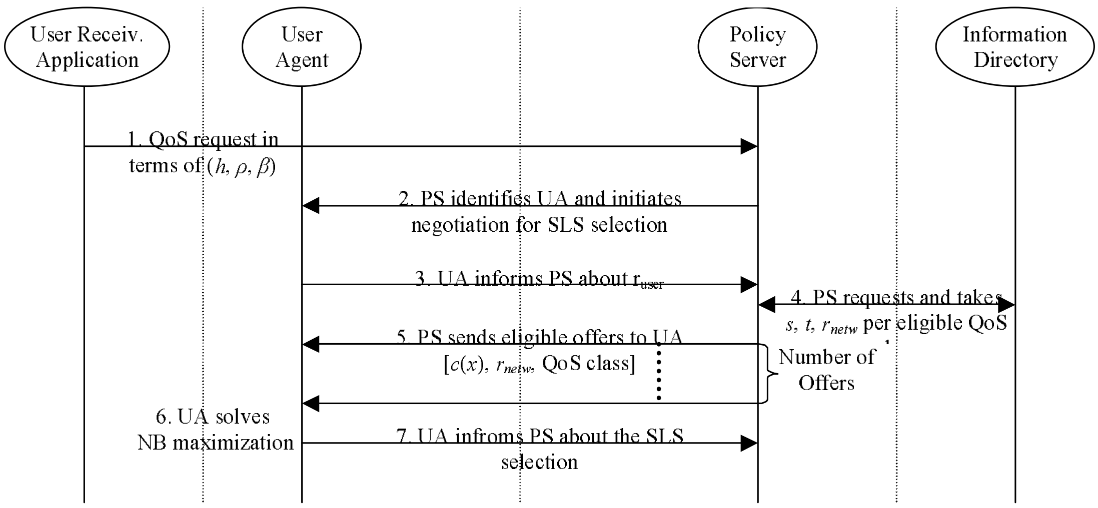


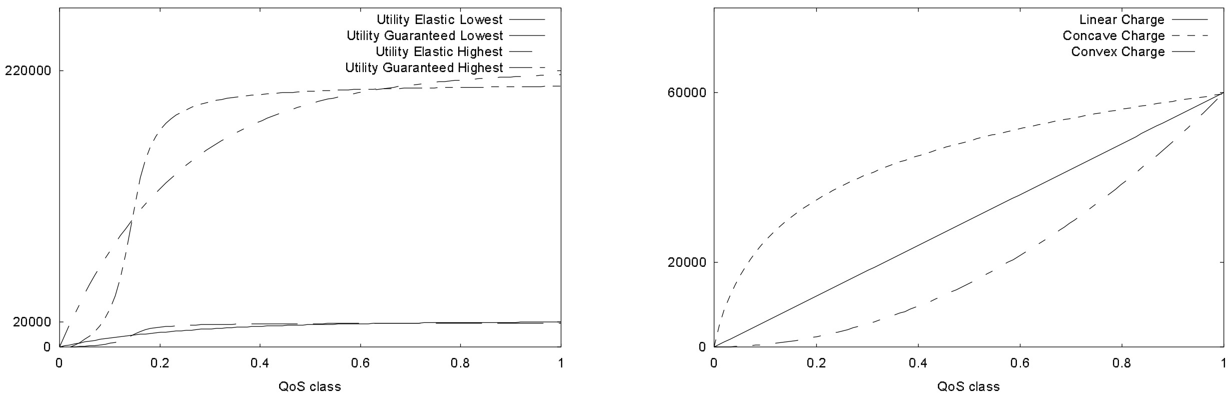
Figure 4. The SLS selection process.

7. EXPERIMENTAL RESULTS ON ECONOMIC EFFICIENCY

Recall that as described in Section 3, the charging scheme that we use provides the right incentives to users for SLS selection that reflects their actual requirements. However, is incentive compatibility maintained for certain users that apply this negotiation process to select their SLSs? That is, is social welfare (i.e. the sum of user utilities) also promoted when each user performs individual optimization?

To investigate this we performed the following experiment: We compute the social welfare resulting by employing the negotiation process for sharing an amount of resources K to a number of users N, and compare it with the social welfare resulting when sharing the same amount of resources equally to the N users. In this experiment, all users request the same service and content (e.g. the same video). Half of the users are taken as elastic and the rest of them as guaranteed. Furthermore, the utility functions are parameterized and the users have different willingness to pay. All parameters and willingness to pay follow uniform distributions of various means and variances (the exact parameterization and the distributions of the various parameters as well as of the willingness to pay are discussed below). The net benefit maximization process selects a certain QoS class (and, thus, resource consumption) for each user to be served. These selections also determine the total resources K that should be allocated and the total charge C_{tot} for the users. The charging function $c(k)$ of the network for a QoS class k, using the charging scheme we mentioned, can have any increasing shape i.e. linear (e.g. if the parameter of QoS class differentiation is bandwidth), convex (e.g. if the parameter of QoS class differentiation is cell loss probability), or concave (according to a specific policy) or a mix of them. We compute the social welfare U_{NB} emerging under the negotiation process.

Then, we share the resources K equally to each user, by offering to all of them the QoS class charged for C_{tot}/N. This corresponds to resource usage K/N for each user. Adding the new utility values for the new QoS class of the same users (i.e. maintaining the same utility functions as previously), we obtain the new total utility U_{fair}. We noticed that $U_{NB} > U_{fair}$ for all the shapes of the charging function and for various uniform distributions of willingness to pay. This advocates that incentive compatibility is maintained with the proposed negotiation process.



(a) **(b)**

Figure 5. The range of the utility functions of guaranteed and elastic users (a), and (b) the various shapes of the charging function (concave, linear, convex).

In particular, we used the following formulae in our experiments:

$$
\begin{aligned}
&U_{elastic}(x) = \left(1 - e^{-\lambda x}\right)W_{max} \\
&U_{guaranteed}(x) = \left\{\left[\tan^{-1}(30x - \lambda) + 1.3\right]/3\right\}W_{max} \\
&C_{linear}(x) = xC_{max} \\
&C_{concave}(x) = \left[\log(1 + 35x)/3.6\right]C_{max} \\
&C_{convex}(x) = x^2 C_{max} \\
&W_{max} = W_{var} + W_c
\end{aligned}
\tag{4}
$$

where x is the QoS class of an SLS, and λ is a random variable that is uniformly distributed in the interval [2.25, 10]. W_{max} is the willingness to pay expressed as the sum of a fixed term W_c = 2000 and a term W_{var}, which is a random variable that is uniformly distributed in the interval [0, 2E[W_{var}]]. E[W_{var}] is fixed (for all simulated users) in each experiment, and in successive experiments takes the values 30000 , 35000, …,100000. Also, C_{max} = 60000 is the charge for the service in the highest QoS class. Figure 6 depicts the percentage of difference in the social welfare attained using the net benefit maximization (U_{NB}) and that using the equal split (fair process) of network resources (U_{fair}), for the three shapes of charging function and for E[W_{var}] ranging between 30000 and 100000. (A continuous set of QoS classes was assumed.)

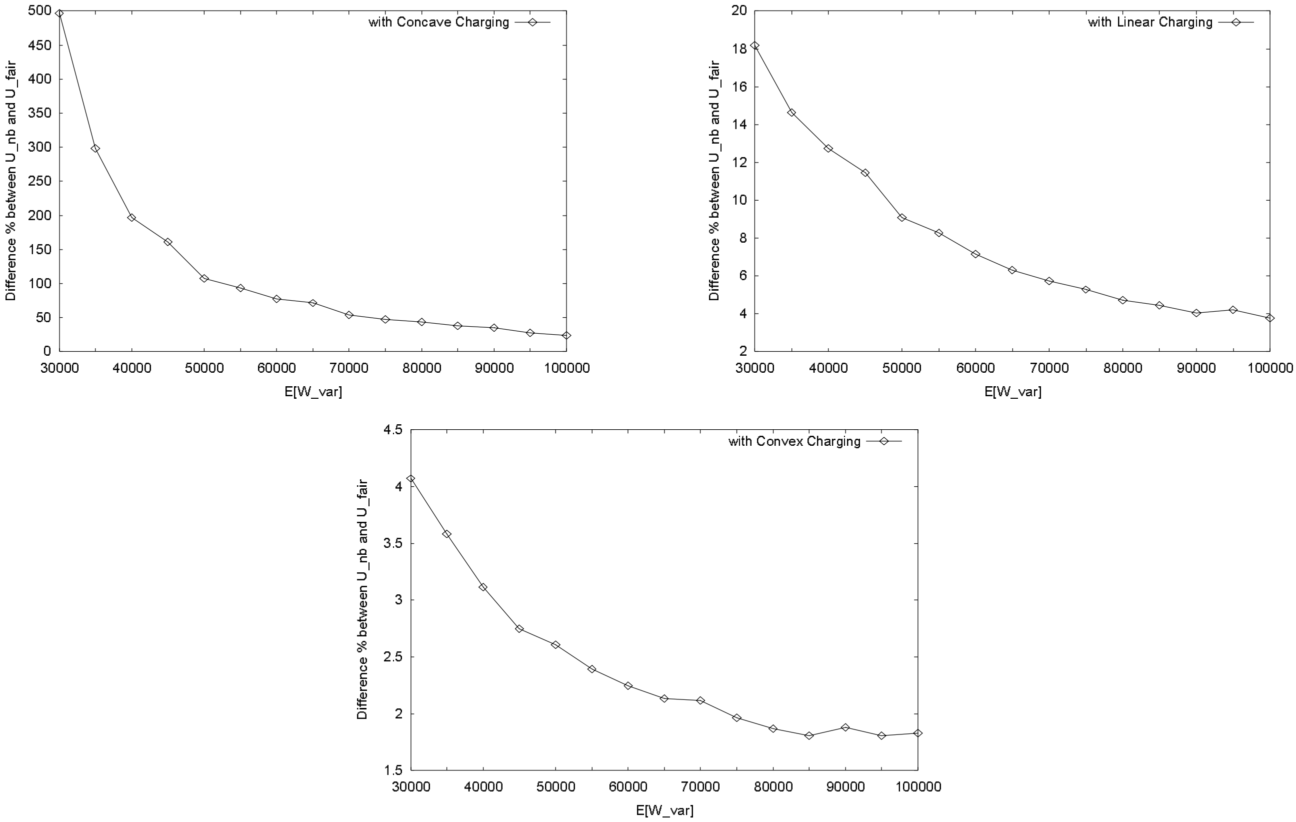


Figure 6. The percentage of difference between U_{NB} and U_{fair}, for the various shapes of the charging function, when E[W_{var}] ranges from 30000 to 100000, and W_c= 20000.

In Figure 6, we observe that the percentage differences between U_{NB} and U_{fair} are always positive, and that their shapes are convex in E[W_{var}]. Therefore, our approach is always more efficient than the fair process. The improvement achieved using our SLS selection process in social welfare is substantial in the cases of concave and linear charge. On the other hand, under convex charging, the mean QoS class costs less than the mean charge, implying that the fair process results in a better QoS class than the mean. Therefore, the loss in social welfare due to the fair process is limited. It should also be noted that when E[W_{var}] is large, then almost all of the users select a high QoS class. Thus, the fair process does not alter significantly the SLS of the most of the users, and the associated loss in social welfare is very small. We have performed the above experiments for various values of the parameters and all results are in favor of incentive compatibility when using the net benefit maximization process to select the optimal SLSs for the users.

8. IMPLEMENTATION AND EVALUATION

So far, we have described the architecture, and the SLS selection approach. In this section, we present specific choices on the implementation of our approach in a realistic network domain. The network domain consists of three routers and three end-hosts. The routers in the network run Cisco MPLS implementation, while the end-hosts run Windows 2000. RSVP was employed for communicating the resource allocation requests, because it provides a clear way for applications to express their QoS requirements. Windows 2000 platform supports RSVP and performs the shaping, policing and marking of the flows according to RSVP signaling. Furthermore, according to the process described in Section 4, the computation of the RSVP traffic parameters for the resource reservation request can be transparent for the end-user, who may ask for a required QoS class in an abstract way (e.g. adjusting a slide bar, thus asking for a more expensive and better service or for a less expensive and inferior one). In the ingress routers, we used the protocol Common Open Policy Service (COPS)[11., 12.] in order to intercept RESV messages and forward them to the Policy Server, which is partly a COPS server (Figure 7). Recall that the PS location is not known to the UA. The choice of COPS was preferred from other eligible protocols (SNMP, TelnetCLI), because it facilitates modifications of reservation requests according to the results of the negotiation process as well as discovery of PS location. The Policy Directory (PD) and the Information Database (ID) are implemented using Microsoft Active Directory. The PS uses LDAP for the storage and retrieval of information to PD and ID.

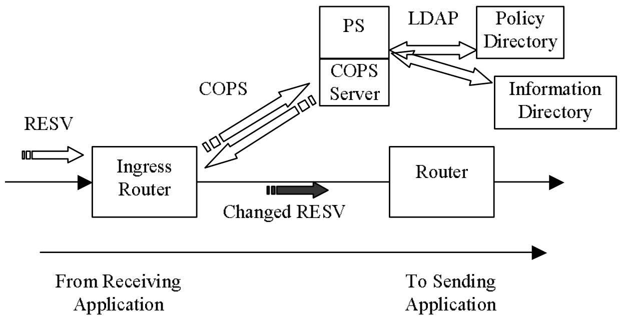


Figure 7. The use of COPS protocol to intercept QoS request and employ QoS provision according to the SLS selection process.

A user sends a RESV message to an application server with the QoS request for a new flow reception. Upon reception of the RESV message from an ingress router of our network, the following take place:

1. The RESV message is redirected to the PS.
2. The PS using the destination address contained in the SESSION object of the RESV message identifies the user's UA and negotiates with it.
3. A specific SLS is selected by the negotiation and the PS uses COPS in order to change the RESV message properly. The new RESV message contains the traffic parameters with which the flow is served in the selected QoS class and a DCLASS object with the DSCP of the QoS class.
4. The RESV message passes transparently through the other routers of the network, i.e the RSVP protocol is used only for signaling purposes and not for resource allocation.
5. The application server, receiving the RESV message, sets the shaping, policing and marking rules for the new flow transmission. The specific DSCP contained in the DCLASS object of the received RESV message is used by the server for marking the packets of the new flow.

Note that the use of RSVP along with the COPS protocol provides a means of controlling resource usage according to the results of the negotiation process. All resource requests are forwarded to PS, which modifying them according to the results of the negotiation, controls the resource allocation procedure in a network domain. The PHBs are applied to the traffic entering the MPLS network according to the DS values of the packets using the E-LSP method described in Ref. 7.. According to the E-LSP method, the information about the PHB to be applied by the network is conveyed to the EXP field of the Shim Header of the MPLS packets, for up to eight traffic aggregates along a LSP. Finally, the state information for the SLSs and the traffic aggregates in the various PHBs are stored in a Microsoft Active Directory using LDAP.

The proposed architecture can be used as a framework through which a network provider can enforce the results of a SLS selection process, according to certain policies for resource allocation. This is accomplished with minimal processing overhead and modification requirements for the network core, since traffic classification, policing, shaping and marking operations are done by the edge hosts. Furthermore, the signaling overhead is also minimal as every reservation request is processed only once and its state is stored accordingly. Thus, the architecture scales well for large network domains. It is preferable in terms of performance to have multiple instances of PS spread inside the network domain. This way, the processing requirements for a particular PS and the communication overhead of the SLA selection are reduced. Indeed, in this case, the requests are shared among the PSs and the links traversed by the messages exchanged during the negotiation are less. Using our approach for SLA selection, the network can offer the right incentives for individual users. This approach involves the exchange of a few messages only (as many as the number of eligible SLSs). It results in a traffic categorization in a QoS class for a new flow.

9. CONCLUSION AND FUTURE WORK

In this paper we developed and evaluated an efficient SLA selection mechanism for a DiffServ-over-MPLS network domain. Experimental results indicate that users are provided with the right incentives. An important feature of our approach is the simplicity of user's procedure for selecting optimal SLS parameters. We implemented our approach in a real network domain that proved its efficiency in terms of performance. Our approach can also be employed as a complement to the traffic engineering procedures, by modifying the price per unit of effective usage for each QoS class, either dynamically (thus balancing the load among the QoS classes), or statically (thus offering users the incentives to select certain paths instead others). To conclude, our architecture offers to the network provider the flexibility to apply pricing and/or resource allocation policies for improving efficiency in network operation. We plan to enhance and evaluate a number of policies in our testbed. Thus, we will provide an extended intra-domain architecture both for the SLA selection and employment, and for efficient management of the network resources. Another subject of future research is the application of our negotiation process for proper SLA selection between network domains. Finally, we plan to extend our architecture to address end-to-end dynamic SLA provision across DiffServ network domains.

ACKNOWLEDGEMENTS

The authors are grateful to Magda Chatzakis, Costas Courcoubetis, and Vasilios Siris for useful discussions on the subject of the paper.

REFERENCES

1. S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang and W. Weiss, "An Architecture for Differentiated Services", IETF RFC: 2475, December 1998.
2. R. Braden, L. Zhang, S. Berson, S. Herzog and S.Jamin, "RSVP Functional Specification", IETF RFC: 2205, September 1997.
3. C. Courcoubetis and V. Siris, "Managing and Pricing Service Level Agreements for Differentiated Services", In *Proceedings of IEEE/IFIP IWQoS'99,* London, May 31 – June 4, 1999.
4. K. Nichols, V. Jacobson, L. Zhang, "A Two-bit Differentiated Services Architecture for the Internet", IETF RFC: 2638, July 1999.
5. Y. Bernet, R. Yavatkar, P. Ford, F. Baker, L. Zhang, M. Speer, R. Braden, B. Davie, J. Wroclawski and E. Felstaine, "A Framework For Integrated Services Operation Over DiffServ Networks", IETF Internet Draft: <draft-ietf-issll-diffserv-rsvp-05.txt>, May 2000.
6. J. Wroclawski, "The Use of RSVP with IETF Integrated Services", IETF RFC: 2210, September 1997.
7. F. Le Faucheur, S. Davari, P. Vaananen, R. Krishnan, P. Cheval and J. Heinanen, "MPLS Support of Differentiated Services", IETF Internet Draft: <draft-ietf-mpls-diff-ext-07.txt>, March, 2000.
8. G. D. Stamoulis, D. Kalopsikakis, A. Kirikoglou, and C. Courcoubetis, "Efficient Agent-based Negotiation for Telecommunication Services", In *Proceedings of Globecom'99*, Rio de Janeiro, Brazil, December 1999.
9. C. Courcoubetis, F. Kelly and R. Weber, "Measurement-based usage charges in communication networks", Statistical Laboratory Research Report 1997-19, University of Cambridge, 1999.
10. S. Shenker, C. Partridge and R. Guerin, "Specification of Guarranteed Quality of Service", RFC: 2212, September 1997.

11. D. Durham, J. Boyle, R. Cohen, S. Herzog, R. Rajan and A. Sastry, "The COPS Protocol", IETF RFC: 2748, January 2000.
12. S. Herzog, J. Boyle, R. Cohen, D. Durham, R. Rajan and A. Sastry, "COPS Usage for RSVP", IETF RFC: 2749, January 2000.
13. R.Guerin, S. Blake and S. Herzog, "Aggregating RSVP-based QoS Requests", IETF Internet Draft: <draft-guerin-aggreg-RSVP-00.txt>, November 1997.
14. Y. Bernet, "Format of the RSVP DCLASS Object", IETF Internet Draft: <draft-ietf-issll-dclass-01.txt>, October 1999.
15. H. R. Varian. *Microeconomic Analysis - Third Edition*. W. W. Norton & Company, New York, 1978 (reprinted 1984, 1992).
16. E. C. Rosen, A. Viswanathan, R. Callon, "Multiprotocol Label Switching Architecture", IETF Internet Draft: <draft-ietf-mpls-arch-07.txt>, July 2000.