Service-Driven Traffic Engineering for Intradomain Quality of Service Management

Panos Trimintzios, George Pavlou, Paris Flegkas, University of Surrey
Panos Georgatsos, Algonet S.A.
Abolghasem (Hamid) Asgari, Thales Research Ltd.
Eleni Mykoniati, National Technical University of Athens

Abstract

Quality of service delivery in IP networks is an important area for service providers, pointing to new business opportunities for premium quality traffic. While there has been relevant research on traffic engineering for QoS management, the problem has never been addressed through a holistic approach that brings together service management and traffic engineering. In this article we present an integrated approach to intradomain QoS management that brings together two-level service management and traffic engineering approaches, coupled through the concept of the resource provisioning cycle. We present validation results for MPLS-based traffic engineering through both testbed experimentation and simulation; we also present validation results for monitoring through testbed experimentation. This article updates, enhances, and validates the top-level view of this integrated architecture first presented in [1].

Quality of service (QoS) delivery across the Internet provides new business opportunities, but also presents new challenges. Nowadays, QoS-based services are offered on the basis of the service level agreements (SLAs), which set the terms and conditions on behalf of both providers and customers for providing and requesting/accessing services, respectively. The area of service-based QoS provisioning has been extensively studied over the last 10 years, focusing primarily on the aspects of SLA provisioning (fulfillment) and monitoring (assurance). Today, providers do offer QoS-based services, mainly within their domain, based on either overprovisioning or other traffic engineering techniques.

IP differentiated services (DiffServ) is widely seen as the framework to provide QoS in the Internet in a scalable fashion. However, many issues have still not been fully addressed, such as the way per-hop behaviors (PHBs) and per-domain behaviors (PDBs) can be combined to provide end-to-end services. Other key aspects not yet fully addressed are admission control, resource reservation, and the role of management plane functionality and its integration with the control and data planes. Multiprotocol label switching (MPLS) can be used as the underlying technology to support traffic engineering.

Our primary goal is to provide an integrated architecture and associated techniques for automated QoS delivery in a DiffServ-capable IP network. Our approach deals with both service and resource management aspects and the relationships between them. We assume the existence of the basic component for defining QoS-based IP connectivity services, the service level specification (SLS) [2], which is the technical part of an SLA.

The work in [3] was the first to propose the notion of the centralized *bandwidth broker* as the means to support *premium service*. This work was a high-level presentation of the required components such as QoS provisioning and admission control for DiffServ networks. In [4] the authors presented the problem of intradomain provisioning for the purpose of balancing the load on the network. This work suggested a centralized tool, which aims to automatically calculate the routing configuration based on a traffic matrix calculated from measurements. In [5] the authors present a bandwidth broker architecture in a centralized way, where the network nodes do not have to keep any QoS state information. The article concentrates on admission control based on the virtual time reference system proposed by the authors. The work in [6] presents a routing and traffic engineering server in order to enforce the authors' suggested minimum interference routing. A good overview of traffic engineering is given in [7]. To our best knowledge, none of the related work considers both service-related aspects and traffic engineering; also, none of them considers multiple levels in traffic engineering or admission control.

This article is a continuation of the initial functional architecture for QoS provisioning presented in [1]. Here we elaborate on the details and functionality of the necessary components, and present results from simulations and testbed experimentation. We adopt a holistic view of providing QoS-based services: vertically, from QoS services to PHBs, and horizontally, from service request handling to service fulfillment and assurance. We propose a two-level approach to both service management and traffic engineering.

The rest of the article is structured as follows. We describe the overall architecture for intradomain QoS provisioning, focusing on the interactions between service management and traffic engineering. We give details of our two-level service management and traffic engineering approaches, respectively. We describe the monitoring system we designed for support-

0890-8044/03/$17.00 © 2003 IEEE

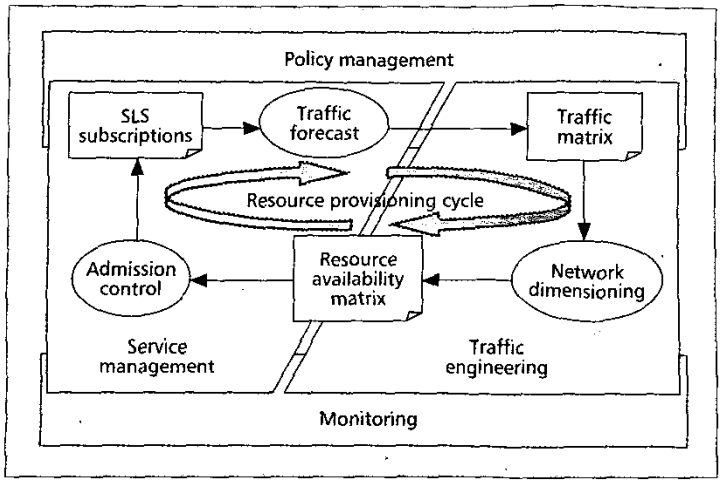


Figure 1. *The overall QoS architecture and the notion of the resource provisioning cycle.*

ing the operation of dynamic traffic engineering and assessing the network and service performance. We discuss the management architecture implementation issues, and finally summarize our findings and discuss directions of future work.

A Holistic Architecture for Intradomain QoS

The QoS architecture we presented in [1] is decomposed, at a high level, into the following major subsystems (Fig. 1):

- *Service management system*, encompassing the service layer functions for service-based QoS service provisioning, which is responsible for agreeing on QoS services with customers and handling respective requests
- *Traffic engineering system*, encompassing the engineering functions of the resource layer, which is responsible for fulfilling the contracted services by appropriately engineering the network
- *Monitoring system*, which is responsible for providing the above systems with the appropriate network measurements and ensuring that the contracted services are indeed delivered at their specified QoS

Recognizing that service provisioning depends largely on business objectives, the *policy management* system is introduced as a means to guide the behavior of the above subsystems in order to address the continuously changing business objectives of the service provider. Policies are defined in a high-level declarative way, and mapped into low-level system parameters and functions. This way, the components' intelligence can be dynamically extended, reduced, or modified through policies. We refer the interested reader to [8] where we describe the policy-based management aspects in detail.

Our architecture distinguishes between *service* and *resource functions*. The resource layer employs network management and traffic engineering functions, while the service layer includes the necessary functions for offering and establishing SLAs (based on the basic SLS template), and subsequently handling the admission of service requests. Although distinguished, the service layer and traffic engineering functions in our system *do not act in isolation*. Traffic engineering functions provide the grounds on which the service layer functions operate and, conversely, the service layer sets the traffic-related objectives [7] for the traffic engineering functions to achieve. More specifically, the service layer provides the traffic matrix to the traffic engineering functions, which specifies anticipated QoS traffic demand between network edges. Traffic demand is also forecast from measured and historical data, current SLS subscriptions, and the service providers' expectations (e.g., sales targets).

Our approach is class-based, and we use the notion of a QoS class, which represents traffic that belongs to a particular PHB and has delay and/or packet loss requirements within a particular range. The entries in the traffic matrix are *traffic trunks*. A traffic trunk represents aggregate traffic that belongs to a particular QoS class and has a certain topological scope. In fact, traffic trunks are aggregates of QoS traffic having the transfer characteristics of the associated QoS class between network edges of the provider's domain.

Resource Provisioning Cycle

As scalability and convergence are of primary concern, interactions and information exchanged between service layer and traffic engineering functions are not fine-grained but rely on aggregate information in order to avoid oscillations and high overhead. As such, the service layer and traffic engineering functions exchange information (traffic and resource availability matrices), which refers to *traffic trunks* and not individual customer contracts or flows, at the granularity of *resource provisioning cycles* (RPCs).

An RPC is a period of time during which anticipated QoS traffic demand for the supported traffic trunks is believed to be valid. Should anticipated QoS traffic demand prove no longer valid, a new RPC is initiated. New RPCs may also be initiated when the service layer or traffic engineering functions realize, each from its own perspective, that they can no longer gracefully provide the QoS being requested. Traffic engineering functions realize this when the network can no longer deliver the QoS targets of the supported QoS classes, and service layer functions when they can no longer satisfactorily sustain the actual offered load at the contracted QoS levels. Note that RPCs may also be initiated by network administration (e.g., when new points of presence or resources are installed).

The interactions between the service layer and traffic engineering functions occur only at RPC epochs, not at the granularity of every single (or few) service request(s). As such, only when a new RPC commences is a traffic matrix produced, the network appropriately engineered, and the resource availability matrix (RAM) produced. By definition, RPCs are relatively long time periods, ranging from hours to days. RPCs are bound to (macro level) traffic forecasts usually drawn from long-term perspectives.

Dynamic Service Management and Traffic Engineering Functions

As anticipated QoS traffic demand and network availability estimates are forecasts, they should be treated as such by the traffic engineering and service layer functions. Actual offered traffic will fluctuate around the forecast values, sometimes even beyond acceptable statistical errors. Our architecture interprets QoS traffic forecasts as nominal rather than actual values, and network availability estimates as guidelines rather than stringent directives. On these grounds, the system incorporates *dynamic* service layer and traffic engineering functions, which, by utilizing actual network state and load information, try to optimize network performance in terms of service admissions and resource utilization while at the same time meeting traffic QoS constraints.

A Simple Example: Virtual Leased Line Service

Let's assume a provider offers a virtual leased line (VLL) service. This could be implemented using the premium QoS class {PHB: expedited forwarding (EF), max delay: 100 ms, max packet loss: 10^{-4}} that maps to a specific DiffServ code point (DSCP) value. Other premium services may also be implemented using the same QoS class.

At the RPC level, based on customer subscriptions to those services that have gone through subscription admission control and on previous usage information, traffic trunks for this QoS class will appear in the traffic matrix. Network dimensioning will produce multiple explicit MPLS paths to support the QoS requirements of those trunks (bandwidth, delay, loss), also optimizing overall network resource utilization. Access nodes will be configured with SLS data (address prefixes), mapping of SLS traffic to particular paths, and conditioning according to SLS traffic profiles. In addition, monitoring activities will be configured for each SLS at relevant ingress and egress node monitors for performance verification.

Traffic of the VLL service will appear at the edge node with DSCP values as agreed. Assuming that the stream is within the agreed performance envelope, it should be accepted and directed to the right path. Service requests beyond the SLS envelope or without any SLS are typically requested through signaling to the edge node, in which case admission control will accept or reject them based on the network state (green, yellow, or red, described later).

A Two-Level Approach to Service Management

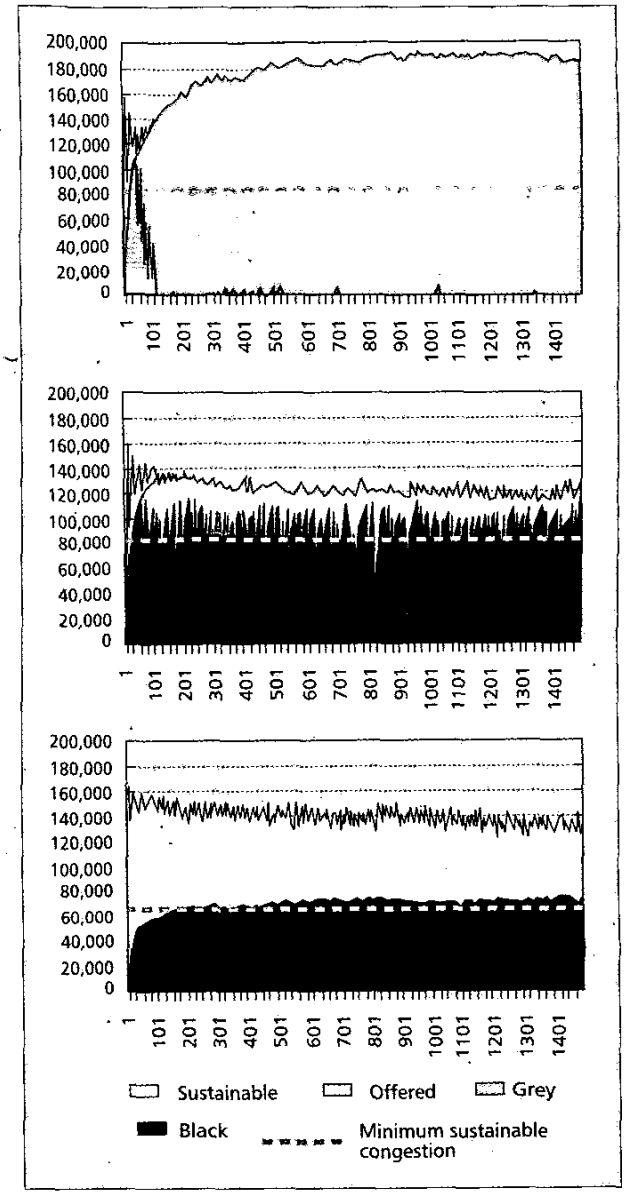
We distinguish two epochs for service requests: *subscription* and *invocation*. At subscription epochs, customers subscribe to services in order to gain the right to use these later. At invocation epochs, customers invoke services to which they have subscribed. Services can be invoked either *explicitly*, using signaling to the edge of the network, or *implicitly*, being active throughout the subscribed service schedule, as are leased line and virtual private network (VPN) services. On successful service invocation, the corresponding traffic flows are generated in the customers' hosts and injected to the network.

The distinction between service subscription and invocation is backed up by a number of facts. It directly corresponds to current business models and practices; is required for authentication, authorization, and accounting (AAA); enables business-driven service provisioning; facilitates traffic demand derivation; and prompts for higher aggregation levels at customer levels, thereby increasing scalability. We include admission logic at both subscription and invocation epochs [9] so as not to overwhelm the network with traffic beyond its current capacity.

Service Subscription Logic

Subscription logic operates during an RPC in a centralized way and utilizes information regarding the resource availability (i.e., RAM) of the engineered network to accommodate QoS services produced by the traffic engineering functions at the beginning of the RPC. Based on this information and related policy parameters, the subscription logic determines for a requested subscription whether to accept it as is, propose alternatives, or reject it.

The objective of the subscription logic is to determine whether service subscription requests can be accepted, in order to avoid eventually overwhelming the network, while maximizing the number of subscribers. We introduce the *satisfaction level* parameter that denotes the degree of satisfaction the service provider opts to provide its subscribers. The system can accept subscriptions either with no traffic considerations at all, or on the grounds of their expected demand, or based on worst-case reasoning, depending on the service provider's policy for this subscription.



□ Figure 2. *Admission control performance results.*

Service Invocation Logic

Invocation logic encompasses two aspects:

- Controlling the number and type of active services
- Controlling the volume of traffic injected by active services

The objectives of the invocation logic are to maximize the number of the admitted services and the QoS they enjoy, thus maximizing network utilization while preventing QoS degradation caused by overwhelming the network with traffic it cannot properly handle.

Invocation logic operates during an RPC and is distributed at the network edges. In order to increase stability and minimize overhead, invocation logic relies on *network state on-off*

indicators rather than network load measurements. Two on-off indicators are considered: *network green* and *network red*, indicating whether or not the network can sustain the performance of the supported QoS classes given the current load. These indicators are set by network monitoring or other SLS assurance services (discussed later). In addition, invocation logic utilizes locally available information on currently admitted QoS traffic in addition to network availability estimates (i.e., the RAM) produced at the beginning of the RPC. Based on this information and related policy parameters, the invocation logic performs admission control through suitable algorithms [9] so as not to overwhelm the network.

Admission Control Results

The proposed admission control functionality was tested for its validity and performance in a computer-based network performance emulation environment, which, by means of a fluid flow model, determines the QoS delivered by the network to the supported QoS classes as a function of their offered load, given the resource availability matrix resulting from the traffic engineering functions. Several tests were conducted considering different network topologies, different offered traffic volumes with respect to the levels against which the network was engineered, and different settings of the policy parameters pertinent to the proposed admission control scheme.

Figure 2 presents indicative results on the performance of the proposed scheme with respect to network goodput and incurred costs. Network goodput — traffic throughput delivered by the network at the agreed QoS — under heavy load conditions is achieved at the cost of subscription acceptance and/or invocation acceptance and/or admitted rate. The light green areas represent network goodput, the darker green areas the traffic delivered by the network with its QoS requirements violated, the gray areas the traffic offered in the network by the accepted invocations, and the black areas the traffic that could be sustained at its agreed QoS.

The first graph corresponds to the case where our scheme is not in effect; all subscriptions and invocations are admitted at their contractual rate, resulting in high admitted traffic volumes and low network goodput. In the second graph, admission control is exerted at invocation epochs only; all subscriptions are admitted, and by means of our scheme network goodput is increased over the previous case, but at the expense of invocation acceptance and admitted rate. In the last graph, admission control is also exerted at subscription epochs, resulting in moderated demand at the invocation level, such as the network can gracefully accommodate, reducing the cost in invocation rejections and rate reductions.

As indicated above and through all experiments conducted, it has been verified that network goodput indeed increases when our scheme is in effect, and the cost incurred depends on the epoch at which our scheme is exerted and the policy parameters tuning its operation.

A Two-Level Approach to Traffic Engineering

We propose a two-level traffic engineering system operating at both long-to-medium and medium-to-short timescales. At long-to-medium timescale, network dimensioning maps the traffic requirements to the physical network resources and provides dimensioning directives in order to accommodate the predicted traffic demands. At the medium-to-short timescale, we manage the routing processes in the network, performing dynamic load balancing over multiple edge-to-edge paths, and ensure that link capacity is appropriately distributed among the PHBs in each link by appropriately selecting the scheduling discipline and buffer management parameters.

The above approach to service-driven traffic engineering can also be referred to as *first plan, then take care*, whereby decisions made with long-term perspectives (beginning of RPCs) are refined by dynamic functions according to actual developments in shorter time periods. The initial decisions and their refinements are driven by clearly defined objectives to meet QoS requirements and optimize network performance. This approach is not monolithic, like pure centralized schemes, while at the same time it harnesses the dynamics of state/load-dependent schemes on the basis of guidelines produced with a longer-term vision. Thus, the provisioning of the network is effectively achieved by taking into account both the long-term service level subscriptions in a time-dependent manner and the dynamic network conditions that are state-dependent.

We argue that both levels are necessary when considering an effective traffic engineering solution, since independently have shortfalls (*centralized* vs. *distributed*, *time-* vs. *state-dependent*, *static* vs. *dynamic*), which can only be overcome when they are used in conjunction.

Network Dimensioning

Network dimensioning performs the long- to medium-term configuration of the network resources. By configuration we mean:

- The definition of label switched paths (LSPs)
- The anticipated loading for each PHB on all interfaces

Dimensioning does not provide absolute values, but they are in the form of ranges, constituting directives for the function of the PHBs (done by resource management), while for LSPs they are in the form of edge-to-edge multiple alternative paths to enable multipath load balancing (done by route management). Note that LSPs are used only as a means of explicit routing, and there is no bandwidth associated with them. Bandwidth is only assigned at the QoS class level.

The *primary objective* of such an allocation is to ensure that the requirements of each traffic trunk are met. The design objectives are further refined to incorporate other requirements, such as:

- Avoiding overloading parts of the network while other parts are underloaded
- Providing overall low network load (cost)

We simplify the optimization problem by transforming the delay and loss requirements to constraints on the maximum hop count for each traffic trunk. This transformation is possible using statistics about the delay and packet loss of the PHBs per link to derive a maximum hop count constraint. For each traffic trunk we seek to minimize the total cost so that the hop-count constraints are met. We use an iterative method where we start from an initial solution, which in our case corresponds to the shortest path solution with link metric inversely proportional to the link capacity, and in each iteration we improve the objective function. The details of this method can be found in [10].

Route and Resource Management

Route management is distributed, operating at each edge router. It is responsible for managing the routing processes in the network according to the guidelines provided by network dimensioning. This includes:

- Setting up routing parameters so that incoming traffic is routed to LSPs proportionally to the bandwidth determined by network dimensioning
- Modifying the routing of traffic according to feedback received from node and network monitoring

During initialization, network dimensioning provides route management with the set of traffic trunks with associated

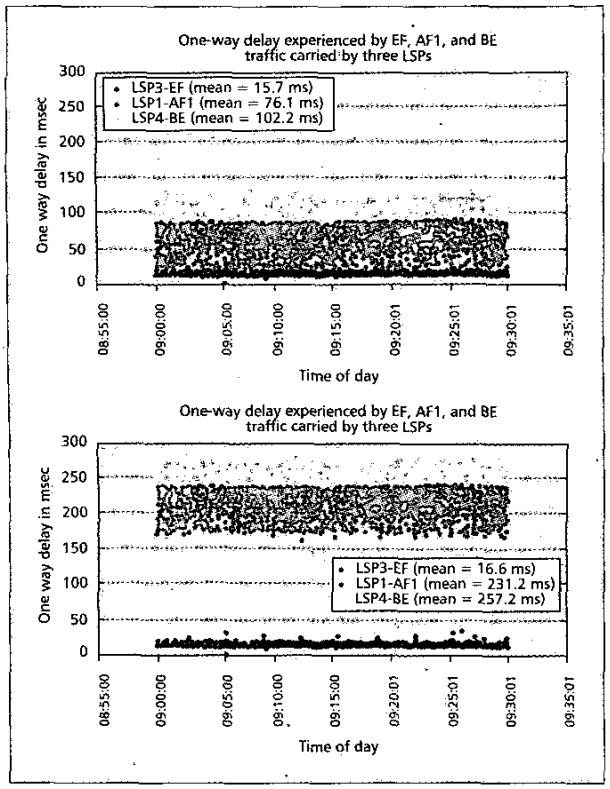


Figure 3. *One-way delay experienced by EF, AF1, and BE LSPs with and without introducing excessive link delay.*

bandwidth and the corresponding LSPs that support this traffic trunk, emanating from the corresponding edge router. During system operation, the QoS performance (end-to-end delay, loss rate, and used bandwidth) of the traffic routed through the LSPs and the PHBs used along the path is monitored. Proactively, we avoid routing traffic to LSPs using the PHBs whose QoS performance in terms of delay and loss performance becomes critical. Hence, actions at this stage attempt to avoid deterioration of end-to-end QoS performance and also help relieve the load on the congested PHBs. Also, route management acts reactively and avoids routing traffic on LSPs whose QoS performance is critical. During this process, mechanisms are employed to ensure that during flow reassignment, the packets-in-order condition is satisfied.

Resource management is distributed at each node. It aims to ensure that link capacity is appropriately distributed between the different PHBs on the link according to dimensioning directives by setting relevant buffer and scheduling parameters. By setting appropriately how the link capacity is partitioned between the several queues and if they can make use of any "unused" capacity or act in isolation, as in hierarchical scheduling disciplines, we can achieve the performance targets set by network dimensioning at the begging of the RPC. The buffer size per PHB is calculated so that the maximum delay for a PHB is preserved, given the service rate allocated to it.

Experimentation and Simulation Results

This section presents some assessment results of our traffic engineering system both on a real testbed composed of commercial routers and in a simulated environment.

In one of the testbed experiments we have defined three QoS classes: expedited forwarding (EF) with delay requirement less than 150 ms, assured forwarding (AF1) with delay requirement less than 400 ms, and best effort (BE) with no delay requirements. Based on the network physical topology and customer site locations, we specified a set of contracted SLSs that were aggregated into traffic trunks. We used our traffic engineering system to automatically configure the testbed taking into account the traffic forecast, and then we introduced excessive delay in one of the links using a data channel simulator. This information is taken into account by network dimensioning, which engineers the network so that end-to-end delay requirements are still met. We can observe from Fig. 3 that both with and without the introduction of excessive delay, the provisioning of the system managed to achieve the delay requirements of all QoS classes. In general, even when dimensioning gets errs with respect to anticipated traffic, monitored values for delay and packet loss are used in the next RPC and the new configuration through this feedback potentially performs better.

In a simulated environment we checked the performance of the traffic engineering algorithms by using various, even very large (up to 300 nodes), random topologies and various loading scenarios. We observed that in all cases the solution provided by our system managed to reduce the maximum link load to feasible levels (Fig. 4). It should be noted that the solution we got by using the shortest paths with link costs inversely proportional to link utilizations was well above link capacities even for relatively lightly loaded scenarios. The execution time of our algorithm was reasonable even for very large networks; for example, for 300 nodes the average running time was below 20 min on a medium-capacity Sun Solaris workstation.

Monitoring for Service Management and Traffic Engineering

Our state-dependent dynamic traffic engineering functions require the observation of the state of the network through a monitoring system in order to apply control actions to drive the network to a desired state. A monitoring system should provide information for the following three categories of tasks:

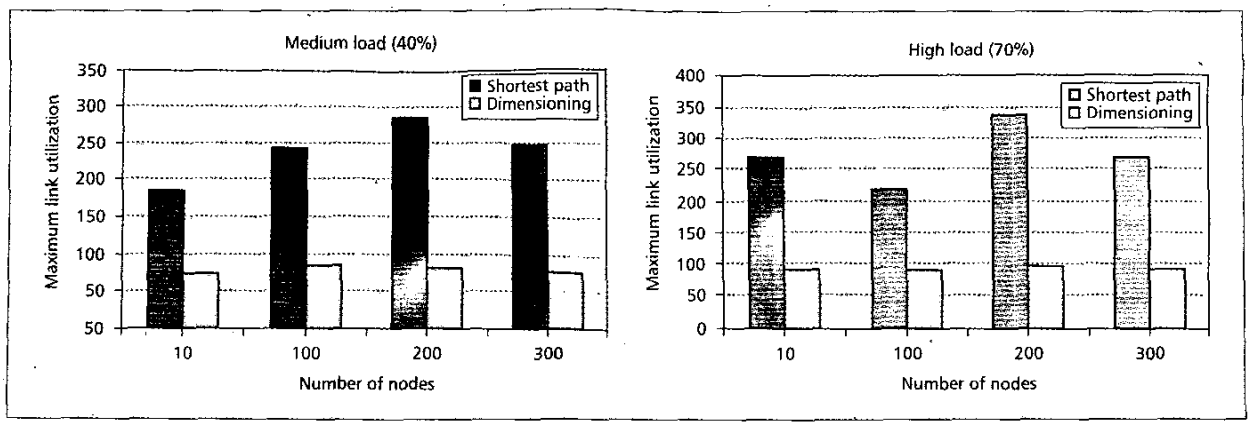
- Assist traffic engineering in making provisioning decisions for optimizing the use of network resources according to short- to medium-term changes (near-real-time monitoring).
- Assist traffic engineering in providing analyzed traffic and performance information for long-term planning in order to optimize network usage and avoid undesirable conditions.
- Verify whether the QoS performance guarantees committed in SLSs are in fact being met.

We view traffic engineering as a continual and iterative process of network performance improvement. The optimization objectives may change over time as new requirements and policies are imposed, so the monitoring system must be generic enough to cope with such changes.

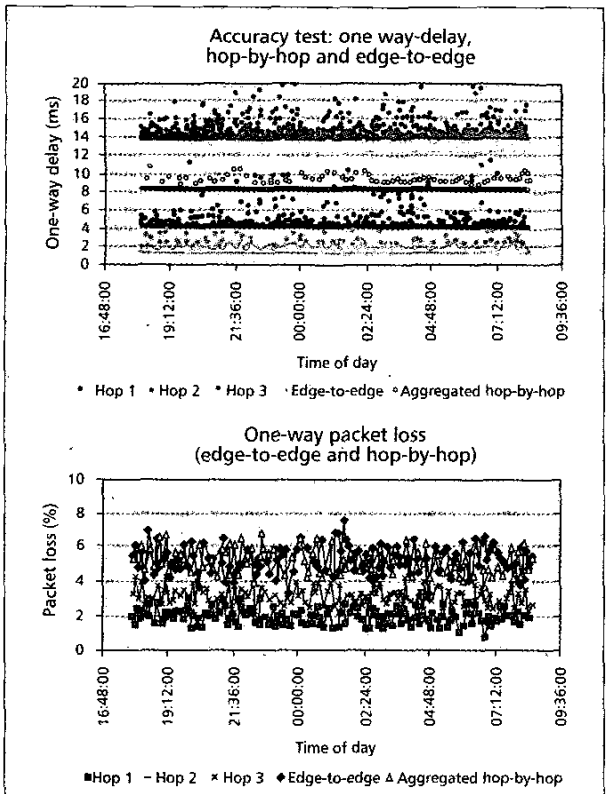
Monitoring System Architecture

Monitoring large-scale traffic engineered networks is a difficult task and requires mechanisms for data collection from a variety of network nodes, aggregation of heterogeneous data sets, data mining of large data sets, and data analysis to generate results for providing feedback to other functional subsystems that require monitoring information. We designed a hierarchical monitoring system to cooperate with the rest of our subsystems, which has the following components:

Node monitor: Responsible for node-related measurements; there is one node monitor per router. The node moni-



■ Figure 4. *Maximum link load for shortest path and dimensioning for medium and high loading scenarios.*



■ Figure 5. *Edge-to-edge vs. concatenated hop-by-hop delay and packet loss results.*

tor is hosted outside of the router on a dedicated workstation, as the availability of required measurements is limited in currently available commercial routers. A node monitor is able to perform active measurements between itself and any other mode monitor at the path or hop level, as well as passive monitoring of the router to which it is attached. It collects measurement results from meters or probes located at routers through *passive* or *active* monitoring agents.

Network monitor: Responsible for network-wide post-processing of measurement data using a library of statistical functions. It is centralized and utilizes network-wide performance and traffic measurements collected by all the node monitor entities in order to build a physical and logical network view (i.e., the view of the routes, LSPs, that have been established over the network).

Service monitor: Responsible for customer-related service monitoring, auditing, and reporting. The SLS monitor is centralized, since it must keep track of compliance in level of service provided to customer SLSs. It utilizes information provided by the network monitor and the various node monitors. The SLS monitor handles the requests for activation or deactivation of monitoring a particular set of SLSs.

Monitoring System Results

The one-way delay results shown in Fig. 3 were taken on our testbed by facilitating the monitoring system. In this section we present additional results that demonstrate the scalability of the monitoring system.

One of the main advantages of our monitoring system is that it only measures hop-by-hop metrics per QoS class. By concatenating individual hop-by-hop measurements over the required path, we deduce edge-to-edge statistics. Thus, we achieve scalability since we only require monitoring per QoS class on every link. This is static information known a priori, rather than monitoring LSPs whose number can be relatively large (on the order of N^2*Q where N is the number of edge nodes and Q the number of QoS classes). Also, this technique helps us to considerably reduce the test traffic introduced to the network in order to perform active measurements for delay and delay variation. Below we look further at the accuracy of hop-by-hop compared with end-to-end measurements.

The experimental results were obtained on a subset of our testbed consisting of four commercial routers connected through three 2 Mb/s serial links in a linear fashion: edge router ER1 connected to core router R1 via link 1; core router R1 connected to core router R2 via link 2; and core router R2 connected to edge router ER2 via link 3. Two data channel simulators (DCSs) were used to introduce delay and loss into links 1 and 3. A commercial traffic generator was connected to both ER1 and ER2, and was used to generate synthetic traffic in a loopback form. The delay results measured by the traffic generator and the packet loss results programmed by the two DCSs were used to verify the results measured by the monitoring system.

Figure 5 shows the delay and packet loss results. For the delay, the average difference between the edge-to-edge and aggregated hop-by-hop measurements is 1.1 ms. This is mainly due to the fact that more processing is required in the hop-by-hop approach, and Ethernet segments connect the node monitors to the actual nodes (i.e., the routers). If active monitoring agents were embedded in the routers, the delay difference would be further reduced. Measuring one-way packet loss over each hop, edge-to-edge, and aggregating hop by hop to

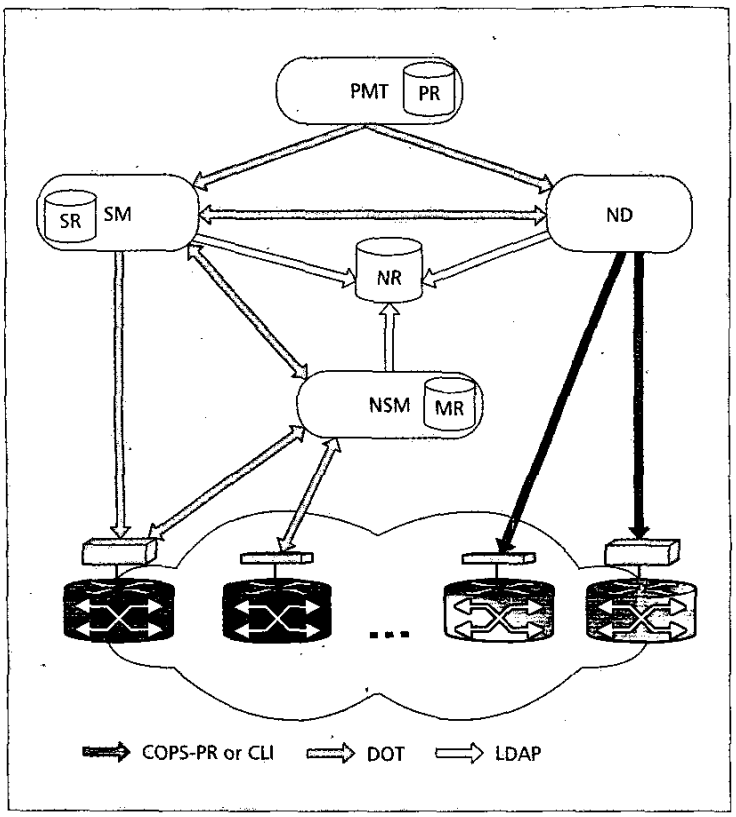


Figure 6. *Management implementation architecture overview.*

deduce edge-to-edge values, the results were 5.26 percent for edge to edge, 2.04 percent for hop 1/link 1, 0.0 percent for hop 2 (there is no DCS on link 2), and 3.19 percent for hop 3, which yields 5.16 percent for aggregated hop-by-hop loss. The difference of 0.1 percent (5.26/5.16 percent) is negligible and can be attributed to rounding errors. Overall, the hop-by-hop method gave comparable results to the edge-to-edge one, having the advantage of enhanced scalability.

Management System Architecture

The architecture of the management system that realizes the described functionality in terms of applications and their interactions is depicted in Fig. 6. The service manager (SM) performs admission control for service subscriptions, holds the service repository (SR), and configures edge routers with SLS related information. The network dimensioner (ND) gets the traffic matrix from the SM and performs offline traffic engineering, producing the logical network configuration and configuring network nodes. Both the SM and ND use the network topology kept in the network repository (NR), although they have different views of it: SM has a service-centric view of edge nodes only, while ND has a complete view of the physical network topology. The network topology is modeled in Common Information Model (CIM) and implemented through a Lightweight Directory Access Protocol (LDAP) database. ND updates NR with the logical topology every time the network is redimensioned. Both SM and ND are policy-driven through the policy management tool (PMT) that provides a graphical policy interface to the human network/service operator (see [8] for details on this).

The network and service monitor (NSM) is a centralized application that configures monitoring activities at each node, and subsequently receives and stores events in the monitoring repository (MR). NSM uses the logical and physical network topology for its monitoring tasks. The interactions with the NR are LDAP-based, while those between any of the other management components are based on distributed object technology (DOT). A Common Object Request Broker Architecture (CORBA) platform has been used in the current implementation, but the interface modeling is general enough to be used with emerging Web-based DOTs such as Simple Object Access Protocol (SOAP)/XML. The configuration interactions to the network are based on Common Open Policy Service for Provisioning (COPS-PR) wherever possible, while DOT interfaces have been used for functionality not yet supported through COPS policy information bases (PIBs).

Every network node consists of the actual label switched router (LSR) and a computing node attached to it that hosts node monitoring, admission control, and route management functionality not currently supported by DiffServ-capable LSRs (the computing nodes are depicted over the switching nodes in the figure). ND configures scheduling and buffering parameters in all nodes through the DiffServ COPS PIB while it passes path information to edge nodes through telnet-CLI (Command Line Interface); the latter then sets up the edge-to-edge paths through signaling. The SM also configures edge nodes with SLS-related data, including conditioning for SLS traffic and associated source addresses. This is also done through CORBA since relevant PIBs do not exist. Finally, the NSM configures node monitors through CORBA, while SNMP is used for local monitoring between the computing station and the LSR.

In summary, we have used existing technologies such as LDAP, COPS-PR, CLI, and SNMP as much as possible, and resorted to more flexible DOTs where relevant standards were absent. The additional functionality DiffServ-capable LSRs should have in order to support our architecture is the following: admission control at the invocation level in order to accept flows conforming to SLSs, flows outside an SLS envelope, and flows without an SLS at all; and route management in edge nodes for load balancing among LSPs supporting a traffic trunk.

Summary and Future Work

The main strengths of our approach to intradomain QoS provisioning can be summarized as follows:

- We specified [2] a formal definition of an SLS template as the basic component used to enable the unambiguous definition of a value-added IP connectivity service.
- The proposed architecture makes a clear distinction between the customer (SLS) and the resource (QoS class) aware components. The interworking is defined through the *resource provisioning cycle*. The service management system has knowledge about all customers but is agnostic about the internal network details, having only a view of the network as a cloud with edge-to-edge capabilities. The traffic engineering system knows about all network resources but operates only on a per aggregate basis. The interaction between these components yields a service-driven traffic engineering solution.
- The architecture introduces a two-level approach for operational service management and negotiation (i.e., service subscription and service invocation). These two processes

occur at different timescales. Subscription handles the longer-term service requests, while service invocation occurs on a per-flow basis, typically within the envelope of a previous subscription.

- The two-level approach in service management is mirrored in the resource management system. The architecture considers relatively long-term network dimensioning based on the requirements of contracted services and subsequent dynamic route and resource management that react in shorter timescales to statistical traffic fluctuations and varying network conditions.
- We propose a scalable monitoring architecture that is able to cope with growing network size and number of customers, and is key to supporting QoS and providing service assurance.
- We validated algorithmic parts of our provisioning approach through simulation, and also built a prototype system operating on a network with commercial routers on which we validated the overall approach through experimental results.

Premium IP services will only see significant uptake when provided across domains in the Internet as a whole. In this context, SLSs between peer providers will be negotiated through service management functions, while actual QoS capabilities may be advertised, (e.g., through extensions to interdomain routing protocols). We are currently focusing our research efforts on interdomain QoS.

Acknowledgments

This work was partially supported by the Commission of the European Union, under the Information Society Technologies (IST) TEQUILA and MESCAL projects. The authors would like to thank their colleagues in the two projects for their useful contributions to the ideas presented in this article, and the anonymous reviewers for their constructive feedback.

References

[1] P. Trimintzios *et al.*, "A Management and Control Architecture for Providing IP Differentiated Services in MPLS-Based Networks," *IEEE Commun. Mag.*, vol. 39, no. 5, May 2001.
[2] D. Goderis *et al.*, "Service Level Specification Semantics and Parameters," IETF draft-tequila-sls-01.txt, Dec. 2001; www.ist-tequia.org/sls
[3] K. Nichols, V. Jacobson, and L. Zhang, "A Two-Bit Differentiated Services Architecture for the Internet," IETF RFC 2638, July 1999.
[4] A. Feldmann and J. Rexford, "IP Network Configuration for Intradomain Traffic Engineering," *IEEE Network*, vol. 15, no. 5, Sept./Oct. 2001, pp. 46–57.
[5] Z.L. Zhang *et al.*, "Decoupling QoS Control from Core Routers: A Novel Bandwidth Broker Architecture for Scalable Support of Guaranteed Services," *Proc. ACM SIGCOMM 2000*, Sweden, Aug. 2000.
[6] P. Aukia *et al.*, "RATES: A Server for MPLS Traffic Engineering," *IEEE Network*, vol. 14, no. 2, Mar./Apr. 2000.
[7] D. Awduche *et al.*, "Overview and Principles of Internet Traffic Engineering," IETF RFC3272, May 2002.
[8] P. Flegkas, P. Trimintzios, and G. Pavlou, "A Policy-based Quality of Service Management Architecture for IP DiffServ Networks," *IEEE Network*, vol. 16, no. 2, Mar./Apr. 2002.
[9] E. Mykoniati *et al.*, "Admission Control for Providing QoS in IP DiffServ Networks: the TEQUILA Approach," *IEEE Commun. Mag.*, vol. 41, no. 1, Jan. 2003.
[10] P. Trimintzios *et al.*, "Quality of Service Provisioning through Traffic Engineering with Applicability to IP-Based Production Networks," to appear, *Comp. Commun.*, vol. 26, no. 9, 2003.

Biographies

PANOS TRIMINTZIOS (p.trimintzios@surrey.ac.uk) holds a B.Sc. in computer science and an M.Sc. in computer networks, both from the Computer Science Department of the University of Crete, Greece. Until 1998 he was research associate at ICS-FORTH, Greece, working on research projects involving high-speed network management and charging. Currently he is a research fellow at the Center for Communication Systems Research (CCSR), University of Surrey, United Kingdom, where he is also completing his Ph.D. His main research interests include traffic engineering, QoS provisioning, policy-based networking, service management, network performance control, and network monitoring.

GEORGE PAVLOU (g.pavlou@surrey.ac.uk) is a professor of communication and information systems at the Center for Communication Systems Research, Department of Electronic Engineering, University of Surrey, where he leads the activities of the Networks Research Group. He holds a diploma in engineering from the National Technical University of Athens (NTUA), Greece, and M.Sc. and Ph.D. degrees in computer science from University College London, United Kingdom. His research interests include network planning and dimensioning, traffic engineering, policy-based networking, programmable/active networks, and multimedia service control. He has contributed to standardization activities in ISO, ITU-T, TMF, OMG and IETF.

PARIS FLEGKAS (p.flegkas@surrey.ac.uk) received a diploma in electrical and computer engineering from Aristotle University, Thessaloniki, Greece and an M.Sc. in telematics (communications and software) from the University of Surrey in 1998 and 1999, respectively. Currently he is studying for his Ph.D. while he is a research fellow in the Networks Research Group at the Center for Communication Systems Research (CCSR) of the University of Surrey, working on EU and U.K. funded research projects. His research interests are in the areas of policy-based management, traffic engineering, IP QoS, and network and service management.

PANOS GEORGATSOS (pgeorgat@egreta.gr) received a B.Sc. degree in mathematics from NTUA in 1985, and a Ph.D. degree in computer science, with specialization in network routing and performance analysis, from Bradford University, United Kingdom, in 1989. He works for Algonet S.A., Athens, Greece, as head of the R&D group. His research interests are in the areas of service management and network performance evaluation.

ABOLGHASEM (HAMID) ASGARI (hamid.asgari@rrl.co.uk) received his B.Sc. from Dr. Beheshti University in Tehran, M.Sc. (Honors) from the University of Auckland, New Zealand, both in computer science, and his Ph.D. in the design and analysis of ATM networks from the University of Wales, Swansea in 1997. He joined Thales Research and Technology, United Kingdom, in 1996, where he is a principal engineer specializing in data communication systems/networks. He has been involved in simulation, design, and analysis of integrated IP services and network architectures. He was active in the IST TEQUILA project, mainly focusing on intradomain QoS monitoring at the network and service levels, and system-level performance evaluation. His research interests are in IP QoS technologies, QoS-aware monitoring, and network performance evaluation.

ELENI MYKONIATI (mykel@telecom.ntua.gr) received a B.Sc. degree in computer science from the University of Piraeus, Greece, in 1996. Since 1997 she has worded as a research associate in the Telecommunications Laboratory at NTUA. In 1998 she became a Ph.D. candidate. Over the last few years she has been involved in the REFORM, VIKING ACTS, and TEQUILA IST research projects. Her main research interest is in the service management area.