Flow-Based QoS Management Architectures for the Next Generation Network

Jinoo Joung, Jongtae Song, and Soon Seok Lee

At the extremes of the complexity-performance plane, there are two exemplary QoS management architectures: Integrated Services (IntServ) and Differentiated Services (DiffServ). IntServ performs ideally but is not scalable. DiffServ is simple enough to be adopted in today's core networks, but without any performance guarantee. Many compromise solutions have been proposed. These schemes, called quasi-stateful IntServ or stateful DiffServ, however, have not attracted much attention due to their inherently compromising natures. Two disruptive flow-based architectures have been recently introduced: the flow-aware network (FAN) and the flow-state-aware network (FSA). FAN's control is implicit without any signaling. FSA's control is even more sophisticated than that of IntServ. In this paper, we survey established QoS architectures, review disruptive architectures, discuss their rationales, and points out their disadvantages. A new QoS management architecture, flow-aggregate-based services (FAbS), is then proposed. The FAbS architecture has two novel building blocks: inter-domain flow aggregation and endpoint implicit admission control.

Keywords: QoS, network architecture, IntServ, DiffServ, flow-based, FAN, FSA, flow-aggregate-based services (FAbS).

Manuscript received July 27, 2007; revised Jan. 25, 2008.

This work was supported by the IT R&D program of MIC/IITA, Rep. of Korea [2006-S-064-01, Research on BcN Network Engineering Technologies].

Jinoo Joung (phone: +82 2 2287 5452, email: jjoung@smu.ac.kr) is with the Department of Computer Science, Sangmyung University, Seoul, Rep. of Korea.

Jongtae Song (email: jsong@etri.re.kr) and Soon Seok Lee (email: sslee@etri.re.kr) are with Broadcasting & Telecommunications Convergence Research Laboratory, ETRI, Daejeon, Rep. of Korea.

I. Introduction

1. General Concept of QoS Management Architecture

Several QoS architectures or QoS management architectures consisting of several building blocks have been proposed to provide QoS to end users in packet networks in terms of delay, delay jitter, or loss probability. These building blocks cover a rather wide variety of schemes, including admission control, policing, shaping, scheduling, queuing, discarding of packets, and traffic engineering, to name a few. While some of such building blocks work well alone, others must be incorporated with other building blocks to achieve their purpose.

In this paper, we briefly review existing QoS architectures, the building blocks they contain, and their evolution. We then introduce a new QoS architecture that is suitable for the next generation network (NGN) framework currently being developed mainly by ITU-T.

2. Definition of Flow and Flow Aggregate

Before going any further, we would like to clarify the concept of *flow*, which is vital for understanding the various QoS architectures and their differences. In this paper, flow, session, or microflow strictly indicate the IP-level flow, which is distinguished by the 5-tuple in the IP packet header, that is, the IP source address, destination address, protocol number, source port number, and destination port number. If a set of packets have the same 5-tuple, then they belong to a single flow. Consequently, in IP networks, the packets in a flow always are on the same path, and the transmission order among them is maintained. Indeed, this definition of flow is adopted by both ITU-T and IETF.

Any set of packets distinguished by any identifier other than the IP 5-tuple is not called a flow. A set of flows, which may be distinguished by its unique identifier, is called a flow aggregate. In this paper, *flow aggregate* generally refers to real-time flows. Flow aggregates may last end-to-end, or they can last for only a single hop. An example of a flow aggregate is the MPLS label switched path (LSP), which is distinguished by its label, which is virtually unique throughout the entire path. An LSP typically lasts edge-to-edge within a multi-protocol label switching (MPLS) domain. Another example of a flow aggregate is the Differentiated Services (DiffServ) class. Interestingly, even if flows of the same class exit in contiguous nodes in a DiffServ network, they are of different flow aggregates. The class, as a flow aggregate, lasts only within a single node.

The most important nature of a flow or a flow aggregate is that it is under a common nodal behavior, such as queuing, scheduling, or admission control. Therefore, the flow or the flow aggregate is the major control target of various QoS architectures.

3. IntServ

IETF has defined two services on IP networks which are collectively called Integrated Services (IntServ): controlled load service and guaranteed rate service [1], [2]. Controlled load service defines a service that approximates the behavior of best-effort service under lightly utilized networks. Guaranteed rate service, which we refer to as IntServ in this paper, guarantees end-to-end QoS by means of reserving, allocating, and providing an amount of predefined resource to each flow or session in each server. Also, signaling for resource reservation, while managing hundreds of thousands of flows in a network node requires a great deal of work. This complexity inhibits the adoption of IntServ-type QoS architectures in real networks.

4. DiffServ

DiffServ [3] is another approach that has been proposed to solve the scalability problem of IntServ. It classifies packets or the flows to which they belong into a number of traffic classes. The packets are marked accordingly at the edge of a network. Therefore, hard work is only necessary at the edge nodes. Classes may be assigned with strict priorities, or a certain amount of bandwidth is provisioned for each class, as is the case with flows in IntServ. With the support from a proper signaling scheme, DiffServ is a highly simplified version of IntServ, where many flows are aggregated into a single class and treated as a whole. It is generally understood that networks with DiffServ architectures can guarantee end-to-end delay for packets of the highest priority class with expedited forwarding per-hop behavior (EF-PHB), provided that either the network is utilized lightly enough or has a loop-free topology, such as a tree topology. The definition of EF-PHB, however, has been slightly revised with the concept of the packet scale rate guarantee because the original EF-PHB, which guarantees a rate at all time scales to the highest traffic class, has been demonstrated to be faulty [4].

In [5], it was concluded that unless the link utilizations are kept under a certain level, the end-to-end delay with EF-PHB infinitely increases. Let us denote the leaky bucket parameters of the data rate and the maximum burst size of a flow i as ρ_i and σ_i respectively. We assume that all the flows with premium service are constrained by leaky bucket parameters. We also assume that the premium service traffic receives strict priority over other traffic. The delay bound obtained in [5] is that only under the condition that $\alpha < 1/(H-1)$,

$$D \le \frac{H}{1-(H-1)\alpha}\tau, \tag{1}$$

for a case with infinite incoming link capacity, where H is the maximum hop count in the network. Here, τ and α are defined by the inequalities $\sum_{j\in F_S} \sigma_j \le \tau r^S$ and $\sum_{j\in F_S} \rho_j \le \alpha r^S$ for any server S in the network, in which there is a set of flows, F_S. The burst allowance level measured in time for their transmission is denoted by τ, and the network utilization is denoted by α. However, as noted in [6], (1) does not take the non-preemptive nature of strict priority servers into consideration; therefore, this was corrected in [6]. The bound obtained in (1) is valid only when α is less than $1/(H-1)$. For example, if the maximum hop count is 11 in a network, then α must be less than 10% for the network to have a theoretical delay bound.

On the other hand, it was shown in [7] that, in networks without loops, the delay bounds for packets with EF-PHB exist regardless of the level of network utilization. These bounds are quadratically proportional to the maximum hop count in heavily utilized networks and are linearly proportional to the maximum hop counts in lightly utilized networks.

5. Need for Advanced QoS Architectures

The restriction on utilization or topology in DiffServ networks is difficult to accommodate in some networks; therefore, many advanced architectures have been proposed for delay sensitive real-time applications. DiffServ over MPLS [8] can be seen as an improvised add-on that cannot overcome the fundamental problem of DiffServ because the traffic engineering function is orthogonal to conventional DiffServ functions. DiffServ-aware traffic engineering would realize a range of service classes by using different under- or over-provisioning ratios per class [9]. As has been indicated in [10], there is no analytical or experimental evidence to support the validity of this approach.

Another notable approach is to aggregate flows selectively [11]-[13] so that effective compromises are achieved between the extremes of the completely flow-state aware IntServ and the unaware DiffServ. A series of implementation practices in this regard have been proposed. A successful and scalable extension of the DiffServ architecture was presented in [12], in which a framework and an implementation method are developed to realize a per-domain packet-scale rate guarantee in the presence of flow aggregation.

Another approach is based on the argument that achieving an absolute performance guarantee is difficult with conventional DiffServ so that only a relative differentiation is meaningful [14]. Some even go further by arguing that in a core network, the traditional approach of requesting, reserving, and allocating a certain rate to a flow or the flow aggregates (that is, a class in DiffServ) is too burdensome and inefficient. Therefore, the flows should not explicitly request a service; rather, they should be implicitly detected by the network and treated accordingly [10].

II. Previous Works

1. Quasi-stateful Flow-Based Architectures or Stateful DiffServ

There has been a tremendous amount of work to reduce the complexity of IntServ, especially its per-flow state maintenance. One way of reducing the complexity is to be aware of state information of flows, without actually maintaining them in core routers. Scalable core, or core-stateless fair queuing (SCORE) [15] emulates IntServ based on the state information written in a packet header. The main idea of SCORE is to have packets carry per-flow state, instead of having core routers maintain per-flow state. SCORE has provided the basis for a series of similar approaches. We do not provide detailed descriptions because they all share a common concept. Fair allocation derivative estimation (FADE) [16] is a fair share estimation algorithm, which enables dynamic resource management. Its purpose is to help achieve fair residual bandwidth allocation and improved network utilization without introducing costly per-flow overhead in core routers. Within DiffServ, interior routers use FADE to estimate flow fair share in the absence of per-flow information. A feedback mechanism brings this information to the corresponding nodes. Edge nodes then condition traffic using the corresponding fair share estimations. Link-based fair aggregation (LBFA) [17] aggregates flows, not on the basis of their class (as is the case with DiffServ), but on their input link. These architectures have their advantages and disadvantages, but it is generally believed that either the complexity is not significantly reduced in comparison with that of IntServ, or the performance is not significantly improved over that of DiffServ.

2. FAN

Flow-aware networking (FAN) proposed by France Telecom [10], [18] is rightly called a disruptive architecture. It takes an approach contrary to the conventional wisdom that more control gives better performance. FAN is based on the observation that with only simple measurement-based flow-level admission control and minimal flow protection network congestion can be effectively handled [19]. It identifies three types of network status regimes according to the level of congestion: transparent, elastic, and overload regimes.

In an elastic regime, FAN enforces fairness (protection of well-behaving flows) by fair queuing, which is similar to the IntServ approach, but without explicit signaling. A *fair rate* is equally applied to all the flows. Computation of the fair rate usually assumes M/D/1 queue. This lack of discrimination among flows, however, may not be acceptable to network operators.

In an overload regime, two mechanisms are used. The first mechanism is an admission control. A flow-based overload control blocks new flows to protect ongoing flows. It adopts an implicit approach (signaling free) by selectively discarding packets at any node where congestion is detected. In this regard, the admission control is strictly measurement-based. The inherent difficulty of characterizing flows or flow aggregates motivates this approach. The FAN architecture avoids the need to specify traffic parameters and uses measurement-based admission control to account precisely for real traffic characteristics. The second mechanism is multi-path traffic engineering. Flow-aware adaptive routing [18] is used to spread traffic over all routes in a light load. It concentrates on the shortest routes in a heavy load. How to partition the flows and spread them is still a question. By not taking the shortest path exclusively, the network load is increased; thus, the overall efficiency suffers.

In summary, in FAN there is no signaling at all. There is no admission control until there is congestion and there is implicit admission control upon congestion. Every flow is assigned the same bandwidth. This is just doing a little more than doing nothing. FAN's rather non-intervening control, while its performance may not match that of highly interfering architectures, at least introduces a new dimension to solving the QoS problem.

3. Flow-State-Aware Architecture

A. QoS Architectural Requirements Suggested by ITU-T

The ITU-T recommendation Y.1221 and its two amendments specify the five following IP transfer capabilities (TC) of NGNs [20]: dedicated bandwidth (DBW) TC, conditionally dedicated BW (CDBW) TC, statistical BW TC, delay-sensitive statistical BW TC, and best-effort TC.

While the DBW transport capability is similar to that of premium service defined in DiffServ architecture, CDBW requires quite a unique service. Contrary to what its name implies, the CDBW TC guarantees bandwidth to IP-based flows, but without guaranteed delivery of every packet within the flows. The intention in introducing CDBW is to consistently focus losses on a small set of flows and to provide congestion notification signals to the receiving end of such flows. By having such flows within the network, the flows with more stringent loss or delay requirements are more securely protected. Statistical BW TCs are also new. Statistically providing bandwidth to flows is still an open and difficult problem.

The Y.1221 recommendation defines each TC to be based on flow controls. The current draft, "Requirements and Framework for E2E QoS Architecture in NGN," mandates the QoS architecture to support per-flow control granularity. A flow is defined in Y.1221 as consisting of packets with the same source IP address, destination IP address, and TOS field. In other contributions, it is defined by IPv4 5-tuples. This implies a QoS framework with a very fine granularity, such as IntServ.

In a pure IntServ guaranteed service (GS), if a rate is specified for a flow, the e2e delay bound can be calculated from a few network parameters. However, QoS classes defined in Y.1541 and other ITU-T contributions suggest the use of DiffServ in core networks [21].

In summary, ITU-T suggests that the management (reservation, measurement, accounting, and so on) target should be flows, while the QoS architecture should be based on DiffServ, especially at the core. Thus, the aggregation of flows seems unavoidable. In the following sections, the flow-state-aware (FSA) architecture and its significance is briefly reviewed. FSA has been proposed by British Telecom, Anagran, and ETRI. It has been recently approved in Recommendation Y.2121 in January 2008, at the ITU-T NGN-GSI SG13 meeting held in Seoul.

B. Service Contexts and Flow Specifications

Probably the most significant contribution of FSA is its elaborate description of flows. Each flow is assigned a set of parameters defining the details and essential characteristics. Parameters for flow specifications currently defined are the following: flow identity, requested rate, preference priority, packet discard priority, and service contexts.

There are four types of service contexts: maximum rate service (MRS), guaranteed rate service (GRS), available rate service (ARS), and variable rate service (VRS). GRS and MRS flows are similar to flows with guaranteed service in IntServ. MRS differs from GRS in that it has the immediate transfer option, which allows a network to accept without admission control. ARS is similar to ATM available bit rate. Flows with ARS service context may modify their data rate, either by the application request or the network demand. VRS is a combination of MRS and ARS, such that it is guaranteed a minimum data rate but can ask for more bandwidth on demand later. This rather complex differentiation of flows affects further flow characterizations.

Flow identity may be defined by IP 5-tuples and DSCP. It may also be defined by the MPLS label. Therefore, the term "flow" in FSA may indicate either IP 5-tuple flows or aggregates of them. For flows aggregates, aggregation end points are able to create aggregate flow identification and notify the next FSA signaling nodes about the aggregate flow identity. Requested rate (RR) is, for MRS and GRS, the mean data rate the flow requests at which it should be served throughout the flow lifetime. For ARS and VRS, RR is the initial mean data rate; ARS may modify this rate later. For VRS, the initial RR is the minimum rate at which the flow should be served. Later, the flow may increase the mean data rate (with the same RR name). Preference priority is the priority for admission decision. It can be used for packet discard decision in some cases. Packet discard priority (known as "flow state" before the January 2007 ITU-T SG13 meeting) can have two different values, namely "discard first" and "discard last." It is used for packet discard decision upon congestion.

C. In-Band Signaling

FSA emphasizes the use of in-band signaling, although it is not a single mandatory signaling method. DiffServ code point (DSCP) has been suggested as a way to recognize in-band signaling packets. Both signaling packets and plain data packets for FSA should be recognizable, although the exact method has not been defined yet. Proxies may signal instead of end-systems. Active flow identification is also possible. Aggregate signaling can be performed at the edge of a network. There are several types of signaling packets, including request, response, confirm, renegotiate, and close. Several indication flags have been defined, including ignore indication for signal aggregation, changing direction indication, and QoS approval indication. An FSA router sets this indication if it has approved a request from a flow. A router may clear this indication to inform the edge that the request has not been approved.

D. *Dynamic Congestion Notification and Data Rate Enforcement*

FSA requires the network to be able to notify the customer premise equipment of the congestion status or the network rate. The network rate is the desired data rate, which is less than the RR. In this regards, the FSA is similar in concept to ATM ABR rate control. The ABR rate control scheme is somewhat similar to SCORE, in the sense that (flow or network) information is carried by packets. The RM cell imposes RTT delayed response to network congestion.

4. Effect of Flow Aggregation

As previously mentioned, the performance impact of aggregating real-time flows has been investigated in various studies. Within IETF, this possibility is represented in the well known selective flow aggregation and reservation aggregation [22], which is similar to ATM VP. Aggregation, as stated in RFC 3175, brings its own challenges. In particular, it reduces the level of isolation between individual flows, implying that one flow may suffer delay from the bursts of another. Synchronization of bursts from different flows may occur. There is evidence [23], however, to suggest that aggregation of flows has no negative effect on the mean delay of the flows, and actually leads to a reduction of delay in the "tail" of the delay distribution (99% percentile delay) for the flows. There are studies that suggest the aggregation also leads to a maximum delay bound reduction within an aggregation region [13]. These benefits of aggregation to some extent offset the loss of strict isolation.

At this point, we will elaborate on what a flow aggregation is. A flow aggregation assigns an indistinguishable identification to different flows so that they can be treated (queued and scheduled) exactly the same in the subsequent nodes or ports. The packets within a queue (therefore, within a flow aggregate) are served in first-in first-out (FIFO) manner. The flows and flow aggregates may be served by a rate-guaranteeing server such as weighted fair queuing (WFQ).

With regard to the performance of flow aggregates, it is generally assumed that flows are constrained by leaky bucket parameters at the entrance of a network, and the sum of the mean data rate of the flows at any link is less than the link capacity. The observations of researchers regarding delay performance with flow aggregation [5]-[7], [11], [13] can be summarized as follows.

- The maximum burst size of a flow, at the entrance of each node, increases linearly with hops.
- If flows are protected by rate-guaranteeing servers, such as generalized processor sharing (GPS) or WFQ schedulers, then the increased burst size does not affect other flows.
- If two or more flows are aggregated at a node into a FIFO, the maximum burst sizes of the flows do have an affect on the delay performance of other flows within the flow aggregate.

Aggregation, when carefully executed, can improve performance in terms of the delay bound and the mean delay within an aggregation region. The problem arises outside the aggregation region. A flow passing through aggregation and de-aggregation can exhibit worse performance than if it had not been put through aggregation and de-aggregation [7]. To be more precise, if the scheduler at every node is of the weighted-fair-queuing type, the maximum burst size of a flow that passes through the aggregation region becomes larger linearly as hops are passed. This larger burst size has a negative effect on the delay of other flows in the next network. Therefore, while the delay within an aggregate is not worse, the flow suffers heavy delay degradation after the de-aggregation. This fact becomes clear when we consider DiffServ networks. In a DiffServ network, flows are aggregated into a class at the output port of the every node then de-aggregated at the input port of the very next node. The consequence is that the delay bound infinitely increases if the network utilization is above a threshold [5].

For example, it was shown in [24] that the maximum end-to-end delay experienced by a packet in a network of latency-rate (LR) servers can be calculated from only the latencies of the individual servers on the path of the flow and the traffic parameters of the flow that generated the packet. More specifically, for a leaky-bucket constrained flow,

$$D_i \le \frac{\sigma_i - L_i}{\rho_i} + \sum_{j=1}^{k} \Theta_j^{S_j},$$

where D_i is the delay of flow i within a network; σ_i and ρ_i are the leaky bucket parameters, namely, the maximum burst size and the mean data rate, respectively; L_i is the maximum packet length of the flow; and $\Theta_j^{S_j}$ is the latency of the server S_j. The preceding equation implies that a series of nodes, which serve the flow or the flow aggregate i with an LR server, can be seen as a single LR server with a latency equal to the sum of the latencies of individual servers in a series. *Latency* is a unique parameter of a server, and can be interpreted as the worst delay the first packet within a backlogged period can experience at the server.

It should be mentioned, however, that these bounds obtained for individual flows or flow aggregates are based on the conservative deterministic assumptions and analysis; therefore, control based upon such assumption can lead to severe under-utilization of network resources. Furthermore, network operators experience difficulty when they have to specify such parameters for real traffic. One rare example can be found in the traffic specification (TSPEC) parameters defined for multimedia flows such as G.729A VoIP or MPEG-4/H.263 video in IEEE 802.11e wireless local area networks. While traffic characterizations using leaky bucket parameters are considered conservative, characterizing a flow aggregate is even more challenging. Aggregated traffic has been shown to be extremely difficult to characterize with succinct parameters, notably due to the self-similarity property [17].

For these reasons, there are also approaches to investigate the behavior and performance of flow aggregates based on statistical assumptions and analysis. There is evidence that the performance with flow aggregation can be satisfactory in many cases. For a flow aggregate with enough flows, as long as the sum of average rates remains less than the link capacity, the statistical delay or loss due to congestion is negligible with non-preemptive priority queuing [25]. This observation opens a new direction for network design. Even with DiffServ expedited forwarding services the performance can be guaranteed within a domain.

III. FAbS Architecture

So far, we have reviewed several QoS architectures including DiffServ, FAN, and FSA. These architectures, if we closely examine them, actually contain schemes to resolve instantaneous congestion, to resolve sustainable congestion, and sometimes to avoid congestion avoidance, which can be summarized as follows:

- IntServ: per-flow weighted fair queuing and per-flow admission control
- DiffServ: per-class (a huge flow aggregate that lasts for a single hop) scheduling, per-flow admission control (or per-class rate limiting), and traffic engineering when collocated with MPLS (DiffServ over MPLS)
- FAN: per-flow fair queuing, implicit admission control without any signaling, and traffic engineering
- FSA: per-flow or per-aggregate weighted fair queuing + selective per-flow discard upon congestion, and admission control + selective per-flow discard upon congestion

Next, we describe the architecture of flow-aggregate-based services (FAbS), which is largely based on FSA flow specifications. The description is divided into three parts, protection from instantaneous congestion, protection from sustainable congestion, and congestion avoidance. Instantaneous congestion refers to packet- or burst-level congestion, while sustainable congestion refers to flow-scale congestion. The distinction between these two is based on insight gained from FAN. Instantaneous congestion occurs in an elastic regime, where only an occasional burst may cause congestion. Sustainable congestion usually occurs in an overload regime, where there are more flows than a network can handle.

1. Resolution of Instantaneous Congestion

A. Protection of Flows with Scalability: Inter-Domain Flow Aggregation

One of the key ideas for providing QoS in NGNs is flow aggregation. The principle behind flow protection in FAbS is still per-flow, or per-aggregate flow scheduling and queuing.

The insight we gain from the observation in section II.3 is that as long as a flow stays in a flow aggregate, the delay bound is only a function of each server's latency. However, when the flow changes its membership of flow aggregates, the burst size becomes the major cause for the delay. In other words, a flow aggregation region should encompass as much of a flow's path as possible. Current NGN architectures generally assume IP-flow-based handling at the edge of each network. This assumption is not a strict requirement, however. For example, ITU-T Y.2111 (Y.RACF) [26] defines the management target at the network-to-network interface to be a session, which can be IP-level flows as well as MPLS- or ATM-level sessions. Therefore, we suggest that it should be possible to aggregate flows across the network domain. We call this mechanism inter-domain flow aggregation (IDFA). The domain is defined as a single administrative network domain such that the flow aggregation policy remains the same within a domain. The algorithmic principles of the IDFA are the following.

1. The starting point of the aggregation region should be as close as possible to the user-network interface (UNI), or more precisely, to the leaky-bucket shaper.
2. The aggregation region should be planned to be as large as possible.
3. While traversing end-to-end, the number of aggregations with new flows should be as small as possible.

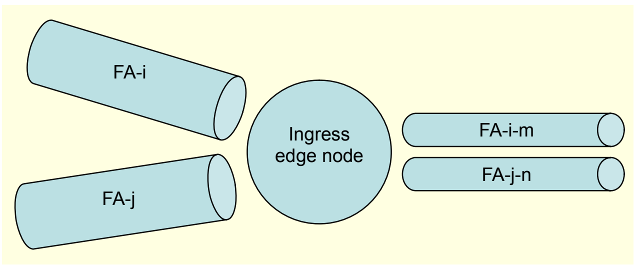


Fig. 1. Possible realization of inter-domain flow aggregation.

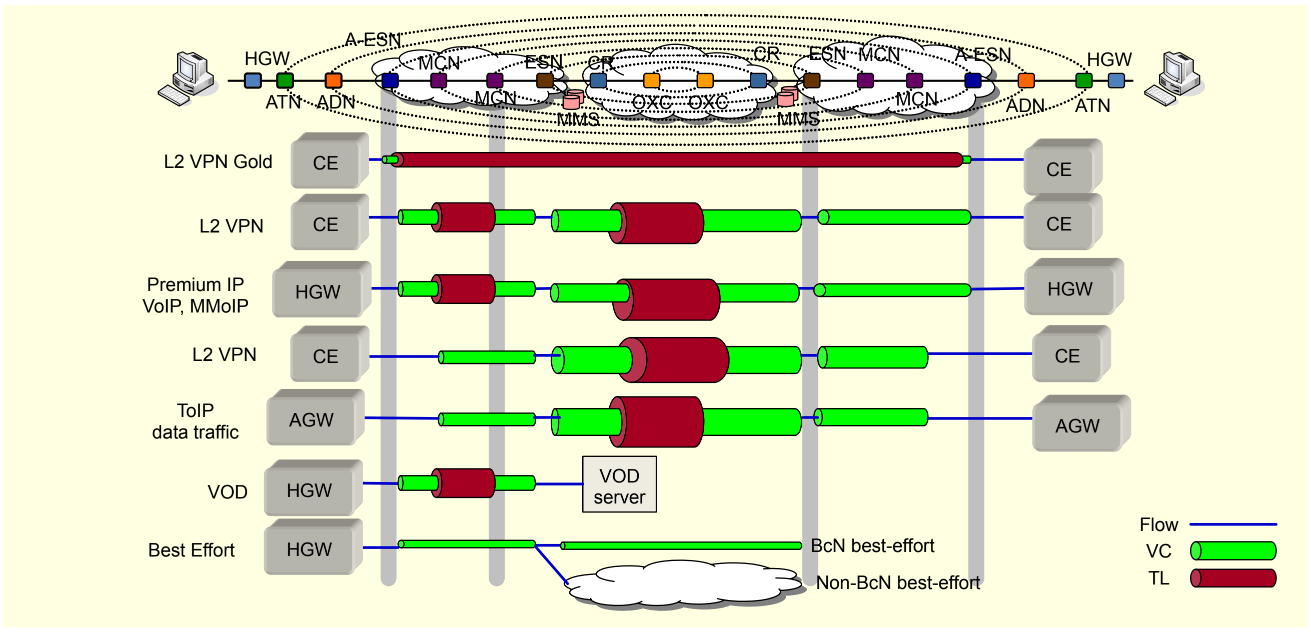


Fig. 2. BcN Architecture.

These principles are illustrated in Fig. 1. If neighboring networks share a common transport mechanism, such as MPLS LSP or ATM virtual channel (VC), then it will be easy to realize the IDFA. The control entities of the networks can exchange the label distribution policy information (in the case of MPLS); therefore, an LSP in a network can be established based on the destination node of the downstream network. Neighboring networks may even employ a single label distribution protocol (LDP). By doing so, these networks can act as a single routing domain.

When seamless flow aggregation is not feasible, partial realization of IDFA still helps to achieve better delay performance. By partial realization we mean aggregation planning under the principles in the previous section. Consider two flow aggregates, FA-i and FA-j, arriving at the edge of the network shown in Fig. 1. Some flows within FA-i and FA-j are destined to the same node in the network. Let us call each sub-flow aggregate FA-i-m and FA-j-n. Without the knowledge of the aggregation policy of the previous network, FA-i-m and FA-j-n are likely to be aggregated in the current network. Instead, with partial IDFA, FA-i-m and FA-j-n are not aggregated. This decision is in conformity with principle 3.

Certainly, an immediate consequence of inter-domain flow aggregation is a larger number of flow aggregates within a network, or equivalently, finer-grained flow aggregates. Therefore, it may not be practical for an aggregation region to encompass the end-to-end path. In such an extreme case, most of the flows may not be able to be aggregated and will simply be treated as individual flows. A brilliant example for such realizations would be the hierarchical flow aggregation of ATM VP-VC or hierarchical LSP by label stacking. Another example worth looking at is the broadband convergence network (BcN) of Korea, depicted in Fig. 2, where end-to-end aggregation is provided for gold service and hierarchical aggregation for other services.

In BcN, the metro and core networks are assumed to support their own routing functionalities. Data traffic is transmitted by the established virtual switched paths (VSPs). A VSP may be implemented in many different forms. In the packet network, VSP is implemented by MPLS and the SDH/SONET networks. The metro and core networks maintain independent VSP structures as well. QoS is guaranteed at the flow level, that is, at the user session level at the edge nodes of VSP. The edge nodes perform flow-level control based on layer-two and layer-three information defined by the IP 5-tuple or other layer 2 header information. Once multiple flows are aggregated in a VSP, the bandwidth per VSP is guaranteed in the transit nodes. For session-based services such as VoIP, packets are transmitted over pre-provisioned VSP. Media gate control and flow-level traffic control are performed at the edge nodes. Flow-level reliability and performance is maintained at the VSP level.

At the edge of core and metro networks, however, the flows are de-aggregated into flows and re-aggregated in the subsequent network. These frequent de-aggregations and aggregations heavily damage the delay performance of a flow, considering the amount of burstiness other flows bring.

The main idea of IDFA is that the flow membership of an FA in one network should remain unaltered, as mush as possible, in the next network. In reality, access networks, metro networks, and core networks are generally of different transport mechanisms. To maintain flow aggregate membership as much as possible, FA identification and its membership information should be handed over to the next network in the data plane or the control plane. When signaling for aggregate management is the limiting factor for scalability, simple in-band signaling can be incorporated by bypassing the MPLS label without popping off at the egress edge of a network.

B. Packet Discard upon Instantaneous Congestion

The FSA architecture defines the priority classes for the CDBW TC described in section II.3. The packet discard priority defined in an FSA network has two basic priorities, namely, discard first and discard last. Any of the four service context flows can be of either priority. FAbS also adopts packet discard priority as a means of packet discard upon instantaneous congestion.

2. Resolution of Sustainable Congestion

Network overload cannot be handled by dropping a few packets or by simply protecting some flows, or both. Flow-level controls are necessary. We propose adopting dynamic admission control and dynamic flow discard to handle the overload or sustainable congestion.

A. Endpoint Implicit Admission Control and Endpoint Rate Limiting with DiffProbe Delay Measurement

The traditional QoS guarantee mechanism, which can be best represented by the guaranteed rate (GR) service of IntServ and the premium service of DiffServ, suffers from two problems: low network utilization and troublesome (therefore, inaccurate) traffic description by the sources. In traditional admission control for hard real-time applications, which require absolute performance guarantee, the simple summation rule is agreed throughout the network. That is, if $\sum$ (data rates) > link capacity, then reject the flow. This admission criteria based on the data rate assignment yields very low network utilization; therefore, it is desirable to increase the utilization by guaranteeing only statistical bounds. To statistically limit the delay or the loss rate, statistical traffic characterization (such as effective envelope) is necessary. Statistical traffic characterization at the source and in the middle of the path, however, is time-consuming, if not impossible. Admission decision based on statistical criteria is difficult as well. Advanced admission control may be applicable. Moreover, many users are unable to describe source traffic parameters *a priori*, except for smooth streaming data. Even encoded video streams, which account for most AV

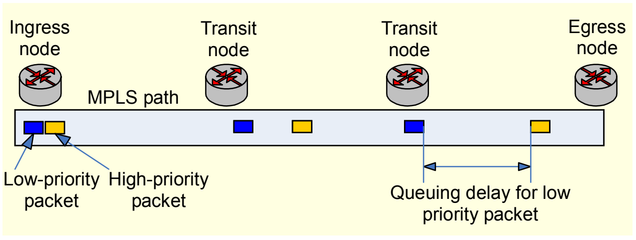


Fig. 3. Concept of DiffProbe.

traffic, are not predictable. For example, in a video conferencing session, the peak rate is not known *a priori*.

A major portion of better-than-best-effort traffic, including ARS service context presumes statistical multiplexing gain. It is very hard to define or justify admission criteria for such a flow aggregate. The implicit admission control in FAN checks for impending overload at every link and discards new flows. If a link is congested, new flows are dropped no matter how far they have traveled in a network. This decreases network efficiency. There must be a way to notify the ingress edge node about the detected congestion. For the detecting node to be able to notify the ingress edge node of the congestion, however, all the network nodes should keep complex mapping tables of flows being served. The suggested admission control in FAbS checks the impending overload using end-to-end delay measurement by DiffProbe [27], which is depicted in Fig. 3. Through DiffProbe signaling, the notified ingress edge of the network can effectively discard new flows without wasting network resource as in FAN. We call this scheme endpoint implicit admission control (EIAC). It is similar to implicit admission control, in that it does not employ the reservation mechanism; but differs in that EIAC takes the control action only at the edges of a network. EIAC also adopts the idea of endpoint admission control (EAC) suggested by Breslau and others [28]. Note that EIAC is only applicable to ARS and VRS service contexts, which take the major portion of the total traffic. For GRS and MRS service context flows, traditional reservation-based admission control should be applied.

DiffProbe is an OAM packet-based delay measurement scheme working in MPLS LSP networks. It measures the one-way delay of the target class (high or low) using the inter-arrival time between the supreme class and the target class. DiffProbe implementation is possible using an ITU-T MPLS OAM framework; however, the precise admission condition using DiffProbe remains to be defined. The admission and call setup procedures should be as simple as possible. In a core network, it is usually not practical to establish an admission policy based on individual flow requests. The complexity of the control procedure can be reduced by utilizing monitoring capability. While the traffic condition is not changed dynamically in the core, the network status should be

Table 1. Comparison of existing QoS architecture and FAbS.

	IntServ	DiffServ	FAN	FSA	FAbS
Resolution of instantenous congestion	Per-flow fair queuing	Per-class (a huge flow aggregate that lasts for a single hop) scheduling	Per-flow fair queuing with an excessively simplified weight assignment	Per-flow or per-aggregate fair queuing+discard upon congestion	Inter-network per-flow or per-aggregate fair queuing+discard upon congestion
Resolution of sustaining congestion	Per-flow admission control	Per-flow admission control (or per-class rate limiting)	Implicit admission control	Admission control+discard upon congestion	Endpoint implicit admission control+discard upon congestion
Congestion avoidance	Not defined	Traffic engineering when collocated with MPLS (e. g. DiffServ over)	Flow-aware adaptive routing	Not defined	Protection switching
Data handling complexity	High	Low	Ideal	Medium (with flow aggregation)	Medium
Signaling complexity	High	Medium	Ideal (non-existing)	High	Medium
Performance	Ideal	Not acceptable	Remains to be seen	Will match that of IntServ	Will match that of IntServ or better

constantly monitored to handle the occasional overloaded situations. Based on the network resource status, service requests are selectively accepted based on their service priority. For example, in under-load states, service requests are accepted without any limitation. When the network becomes congested, only high preference service requests (such as emergency traffic) are accepted.

Any detected congestion through DiffProbe can trigger a rate control mechanism at the endpoint. FSA has defined the rate re-negotiation signaling functions especially in ARS and VRS service contexts. Such a congestion control is being considered for adoption in some current standardization bodies. One notable example is the reactive congestion management proposed in the IEEE 802.1Qau Amendment on VLAN congestion notification [29].

B. Flow Discard upon Sustainable Congestion

The second part of resolving sustainable congestion is flow discard. As in the case of selective packet discard with packet discard priority in FSA, we adopt the preference priority introduced in FSA. Preference priorities may be used by different network operators for the development of different service propositions. When heterogeneous preference priority flows are aggregated into a flow aggregate, an indication of this must be given to downstream nodes. The number of priority levels is network dependent.

3. Congestion Avoidance: Traffic Engineering

Congestion avoidance through traffic engineering is crucial for a QoS management architecture. The term traffic engineering can be arguably interpreted as load balancing and congestion avoidance. Both MPLS LSP and IP tunnel with source routing are exemplary steering handles for traffic engineering. Protection switching, which reserves a backup path for link or node failure or unexpected sustainable congestion can be useful in dynamically resolving congestion. Protection switching, which was devised for optical equipment, is currently used in CISCO routers. It is specified as a standard solution for both IETF and NGN, based on MPLS networks. The recently approved ITU-T recommendation Y.1720 [30] describes the general requirements for protection switching, especially in MPLS networks. It specifies the requirements and mechanisms for 1+1, 1:1, shared mesh, and packet 1+1 protection switching functionality for the user-plane in MPLS layer networks. The mechanism defined in Y.1720 is designed to support end-to-end point-to-point LSPs. However, it does not consider some important problems, including protection switching functionality for multipoint-to-point and point-to-multipoint LSP, M:N protection switching, and hitless protection switching. The fundamental problem of protection switching is that the compromise between network efficiency and bandwidth guarantee is inevitable. If bandwidth has to be guaranteed for a flow, even in the case of failure, another path must be fully reserved to be used by other flows. This greatly reduces network efficiency. Compromises have been proposed, such as M:N backup path and shared mesh. Currently, however, DiffProbe and traffic engineering schemes mentioned in this section are specific to MPLS. Further consideration is necessary for different transport networks, such as the Ethernet.

A comparison of FAbS with the other exemplary architectures is summarized in Table 1. The performance in the table refers to the delay bound and loss ratio guaranteed either statistically or deterministically. The simplicity of FAN cannot be achieved by any other architecture. The performance of FAbS matches that of IntServ. We will further demonstrate this in future work. The different approaches shown in the table are not necessarily exclusive of each other, and some of the approaches may be synergistically used together end-to-end.

IV. Conclusion

We proposed a new QoS architecture called FAbS. The FAbS architecture introduces two novel concepts in QoS management. First, with IDFA, signaling on the data or control planes is exchanged so that the correlation among flow aggregates at contiguous networks is maintained as much as possible. Secondly, EIAC eliminates inefficiency that results from discarding packets in the middle of the path of a flow by congestion notification to the edge nodes. In addition to these novel concepts, FAbS incorporates the best combination of building blocks for different congestion situations. Detailed algorithms for the building blocks will be further developed and proposed as standards in NGN architecture.

References

[1] R. Braden, D. Clark, and S. Shenker, *Integrated Services in the Internet Architecture: An Overview*, IETF RFC 1633, 1994.

[2] P.P. White, "RSVP and Integrated Services in the Internet: A Tutorial," *IEEE Communications Mag*., vol. 35, May 1997, pp. 100-106.

[3] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, *An Architecture for Differentiated Services*, IETF RFC 2475, 1998.

[4] J.C.R. Bennett, K. Benson, A. Charny, W.F. Courtney, and J.-Y. Le Boudec, "Delay Jitter Bounds and Packet Scale Rate Guarantee for Expedited Forwarding," *IEEE/ACM Trans. Netw*., vol. 10, no. 4, Aug. 2002, pp. 529-540.

[5] A. Charny and J.-Y. LeBoudec, "Delay Bounds in a Network with Aggregate Scheduling," *Proc. of First International Workshop of Quality of Future Internet Services (QOFIS2000)*, vol. 1922, 2000, pp. 1-13.

[6] Y. Jiang, "Delay Bounds for a Network of Guaranteed Rate Servers with FIFO," *Computer Networks*, vol. 40, no. 6, 2002, pp. 683-694.

[7] J. Joung, "Feasibility of Supporting Real-Time Traffic in DiffServ Architecture," *Proc. of 5th Int'l Conf. on Wireless/Wired Internet Communications* (WWIC), vol. 4517, May 2007, pp. 189-200.

[8] F. Le Faucheur et al., *Multi-Protocol Label Switching (MPLS) Support of Differentiated Services*, IETF RFC 3270, May 2002.

[9] F. Le Faucheur and W. Lai, *Requirements for Support of Differentiated Services-Aware MPLS Traffic Engineering*, IETF RFC 3564, 2003.

[10] S. Oueslati and J. Roberts, "A New Direction for Quality of Service: Flow-Aware Networking," *Proc. Conference on Next Generation Internet Networks (NGI)*, April 2005, pp. 226-232.

[11] J.A. Cobb, "Preserving Quality of Service Guarantees in Spite of Flow Aggregation," *IEEE/ACM Trans. on Networking*, vol. 10, no. 1, 2002, pp. 43-53.

[12] Y. Jiang, "Per-Domain Packet Scale Rate Guarantee for Expedited Forwarding," *IEEE/ACM Trans. Netw*. vol. 14, no. 3, June 2006, pp. 630-643.

[13] W. Sun and K.G. Shin, "End-to-End Delay Bounds for Traffic Aggregates under Guaranteed-Rate Scheduling Algorithms," *IEEE/ACM Trans. on Networking*, vol. 13, no. 5, Oct. 2005, pp. 1188-1201.

[14] C. Dovrolis, D. Stiliadis, and P. Ramanathan, "Proportional Differentiated Services: Delay Differentiation and Packet Scheduling," *IEEE/ACM Trans. on Networking*, vol. 10, no. 1, Feb. 2002, pp. 109-120.

[15] I. Stoica, S. Shenker, and H. Zhang, "Core-Stateless Fair Queuing: A Scalable Architecture to Approximate Fair Bandwidth Allocations in High Speed Networks," *IEEE/ACM Transactions on Networking*, vol. 11, no. 1, Feb. 2003, pp. 33-46.

[16] N. Li, M. Borrego, and S.-Q Li, "Achieving Per-Flow Fair Rate Allocation within DiffServ," *Fifth IEEE Symposium on Computers and Communications (ISCC),* 2000, p. 340.

[17] Y. Jiang, "Link-Based Fair Aggregation: A Simple Approach to Scalable Support of Per-Flow Service Guarantees," *Networking*, also in *LNCS*, vol. 3042, 2004, pp. 1084-1095.

[18] S. Oueslati and J. Roberts, "Comparing Flow-Aware and Flow-Oblivious Adaptive Routing," *Proc. of CISS*, Princeton, USA, Mar. 2006, pp. 655-660.

[19] T. Bonald, S. Oueslati-Boulahia, and J. Roberts, "IP Traffic and QoS Control," *Proc. of World Telecommunications Congress* (WTC), Sep. 2002.

[20] ITU-T Recommendation Y.1221, *Traffic Control and Congestion Control in IP-Based Networks*, Mar. 2002.

[21] ITU-T Recommendation Y.1541, *Network Performance Objectives for IP-Based Services*, Feb. 2006.

[22] F. Baker et al., *Aggregation of RSVP for IPv4 and IPv6 Reservations*, IETF RFC 3175, Sept. 2001.

[23] D. Clark, S. Shenker, and L. Zhang, "Supporting Real-Time Applications in an Integrated Services Packet Network: Architecture and Mechanism," *Proc. SIGCOMM*, Sept. 1992, pp. 14-26.

[24] D. Stiliadis and A. Varma, "Latency-Rate Servers: A General Model for Analysis of Traffic Scheduling Algorithms," *IEEE/ACM Trans. Networking*, vol. 6, no. 5, Oct. 1998, pp. 611-624.

[25] T. Bonald, A. Proutiere, and J.W. Roberts, "Statistical Performance Guarantees for Streaming Flows Using Expedited Forwarding," *Proc. of IEEE INFOCOM*, vol. 2, 2001, pp. 1104-1112.

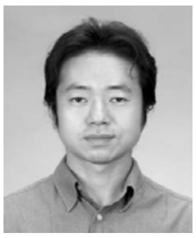
[26] ITU-T Recommendation Y.2111, *Resource and Admission Control Functions in Next Generation Networks*, Sept. 2006.

[27] J. Song, S.S. Lee, and Y.S. Kim, "DiffProbe: One Way Delay Measurement for Asynchronous Network and Control Mechanism in BcN Architecture," *8th International Conference on Advanced Communication Technology (ICACT)*, vol. 1, Feb. 2006, pp. 677-682.

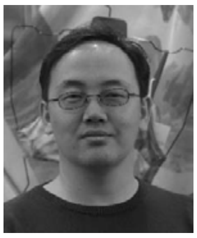
[28] L. Breslau et al., "Endpoint Admission Control: Architectural Issues and Performance," *Proc. of ACM Sigcomm*, vol. 30, 2000, pp. 57-69.

[29] IEEE 802.1Qau Draft 0.1, *IEEE Standard for Local and Metropolitan Area Networks: Virtual Bridged Local Area Networks, Amendment 10: Congestion Notification*, May 2007.

[30] ITU-T Recommendation Y.1720, *Protection Switching in MPLS Networks*, Dec. 2006.



Jinoo Joung received his BS degree in electronics and electrical engineering from Korea Advanced Institute of Science and Technology (KAIST) in 1992, and his MS and PhD degrees in electrical engineering from Polytechnic University, Brooklyn, New York, in 1994 and 1997, respectively. He worked for Samsung Electronics from 1997 to 2005 as the manager of projects developing a Gigabit Ethernet switching ASIC, a packet service module for UMTS, and network processors/communication processors for home and wireless applications. In 2005, he joined the Department of Computer Science of Sangmyung University, Seoul, Rep. of Korea. His research interests include network architecture, QoS control, and various embedded network system designs and implementations.



Jongtae Song is a senior research staff member with Electronics and Telecommunications Research Institute (ETRI), Rep. of Korea. He received his BS degree in electronics and electrical engineering from the Korea Advanced Institute of Science and Technology (KAIST) in 1990, his MS degree in electrical engineering from the University of Southern California in 1994, and his PhD degree in electrical engineering from Polytechnic University, Brooklyn, in 1998. He worked for Bell Labs Lucent Technologies from 1998 to 2001 and for several startup companies (including Coree Networks and Parama Networks) in New Jersey from 2001 to 2004. Since he joined ETRI in 2004, his work has been focused on BcN network architecture, QoS control, and flow-based network control architecture.

Soon Seok Lee received his BS, MS and PhD degrees in industrial engineering from Sungkyunkwan University, Rep. of Korea, in 1988, 1990, and 1993, respectively. In 1993, he joined Electronics and Telecommunications Research Institute (ETRI) where he has worked on several projects related to high-level designing and planning of networks, including ATM networks, mobile networks, optical networks, and so on. In 2003, he served as the chief architect for optical Internet with the Network Technology Lab, ETRI, where he is now a principal member of engineering staff. He is also a project leader for BcN network engineering and high-level design of a network control platform. His research interests include converged network architecture, optical internet, optical networking, network planning and designing, and network performance engineering.