CHAPTER 3

FEEDBACK-BASED QOS CONTROL FRAMEWORK FOR UMTS

Since quality is not essentially guaranteed in IP network, it is important to have proper design and management of networks, terminals and service levels for multimedia services. To have quality design and management, methodologies for appropriately and effectively evaluating user perception of QoS are indispensable. The performances of software applications directly reflect user perceptions of QoS. We propose a QoS feedback framework including a QoS agent as an interface to monitor software performances, which are mapped to user's satisfactions. With this feedback framework, we expect to have a proper QoS control.

In this chapter, we have a brief review on the two main projects launched by Internet Engineering Task Force (IETF) to enhance QoS on IP network: Integrated Services (IntServ) and Differentiated Services (DiffServ).

3.1 Integrated Services (IntServ)

In InetServ, users need to define Traffic Specification (TSpec) that describes the characteristics of packet flow along a certain path, and Requirement Specification (RSpec) for bandwidth reservation through Resource Reservation Setup Protocol (RSVP) (Fig. 3-1). Those routers on the path need to manage RSVP information of path and parameters. Admission control is based on the information of RSVP, resource of routers and bandwidth utilization of links. Each router need to do Multi-Field (MF) classification for identifying,

policing and scheduling each packet flow. In InetServ, all core and edge routers should have the ability to classify, identify and control each packet flow. As the network scale and the amount of packet flows grow up, it will be a challenge for InetServ to handle the overwhelming informing.

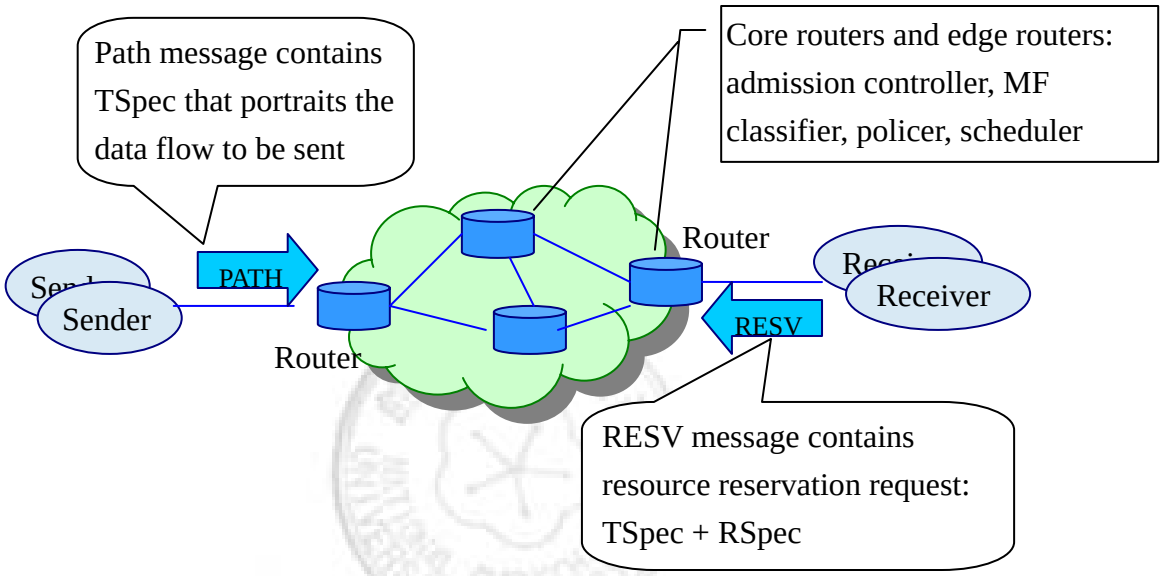

Fig. 3-1 InstServ Network

3.2 Differentiated Services (DiffServ)

Internet Engineering Task Force (IETF) established a DiffServ workgroup in 1998. DiffServ enhancements to the Internet protocol are intended to enable scalable service discrimination in the Internet without the need for per-flow state and signaling at every hop. A variety of services may be built from a small, well-defined set of building blocks which are deployed in network nodes. DiffServ does not provide any service quality guarantee by itself. The services may be either end-to-end or intra-domain, and they include both of those that meet quantitative performance requirements (e.g., peak bandwidth) and those based on relative

performance (e.g., "class" differentiation). Services can be constructed by a combination of distinct “setting bits” in IP header field at network boundaries (autonomous system boundaries, internal administrative boundaries, or hosts).

- Setting bits are used to determine how packets are forwarded by the nodes within a network.
- Packets are marked at network boundaries in accordance to the requirements or rules of each service.

A DiffServ domain could be a Internet Service Provider (ISP). Service Level Agreement (SLA) is signup contract between user and DiffServ domain. The resource will thus be allocated according to SLA. SLA could be changed dynamically. DiffServ simplify the complexity of core router and try to move complex admission control to Bandwidth Broker (BB). If admitted, BB will inform both edge routers and core routers to adjust resource allocation. Packets are classified (MF classification) according to policy at edge router while entering DiffServ domain. The Per-Hop-Behavior (PHB) is then to control flow rate and mark DiffServ Code Point (DSCP), and perform adaptive scheduling.

User subscribed Service Level Agreement (SLA) to service providers. The default SLA contains Traffic Condition Agreement (TCA) which has classification rule, traffic profile and condition rules. Classification rule defines rules about how to classify packets and how to map packets to specific service class. Traffic profile specifies the characteristics of traffic such as flow rate and bust rate. Condition rules define limitations of resource allocation such as bandwidth and throughput threshold.

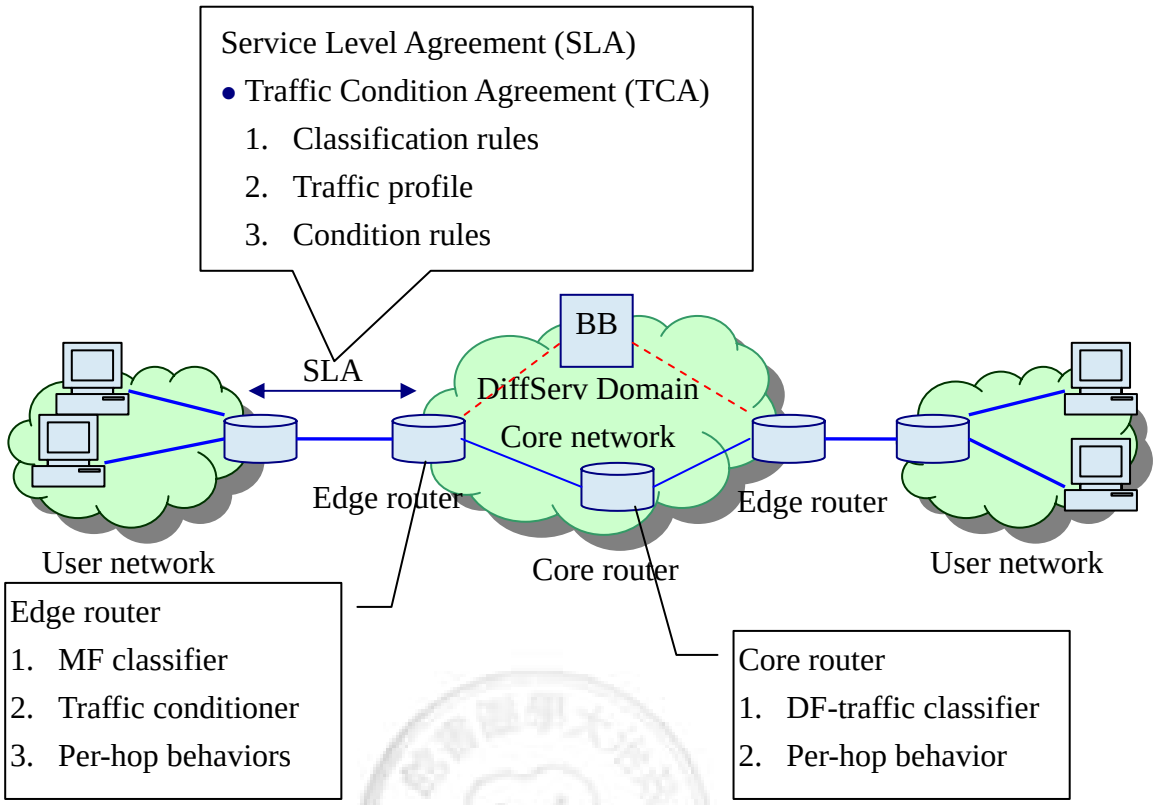

Fig. 3-2 DiffServ Network

Service Level Agreement (SLA) is subscribed between user and providers. The default SLA contains Traffic Condition Agreement (TCA) which has classification rule, traffic profile and condition rules. Classification rule defines rules how to classify packet and mapping to specific service class. Traffic profile specifies the characteristics of traffic such as flow rate and bust rate. Condition rules define limitations of resource allocation such as bandwidth and throughput threshold.

Edge router has three functions which are Multi-Field (MF) classifier, traffic conditioner and per-hop behavior. MF classifier provides the filtering functionality that scans through a variety of packet fields to determine the forwarding class for a packet. Typically, a classifier

performs matching operations on the selected fields against a configured value. Traffic conditioner meters the flow rate to prevent bandwidth exceeding. Per-hop behaviors are different forwarding treatments.

At core router, two functions are provided which are DF-traffic classifier and per-hop behavior. DF-traffic is Default forwarding-traffic which defines default forwarding mechanism. Per-hop behavior is forwarding methods which are EF, AF1, AF2, AF3, AF4 and BE.

The DiffServ architecture contains two main components:

- Simple queue priority mechanisms for forwarding.
- Allocation policies and configuration rules at nodes.

The requirements or rules of each service must be set through administrative policy mechanisms. A differentiated services-compliant network node includes a classifier that selects packets based on the value of the DS field, along with buffer management and packet scheduling mechanisms capable of delivering the specific packet forwarding treatment indicated by the DS field value. Setting of the DS field and conditioning of the temporal behavior of marked packets need only be performed at network boundaries and may vary in complexity. The logical functions of DiffServ are showed in Fig. 3-3

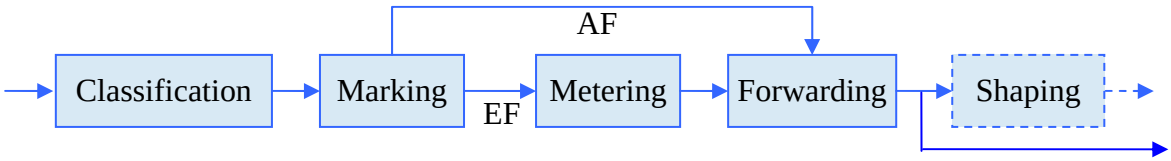

Fig. 3-3 DiffServ Logical Functions

3.2.1 Classification

Classification is classifying packet according to classification rules such as classifying Type of service, Traffic Class, port number or IP address. DiffServ classifies service classes and we add per-flow classification for per-flow QoS guarantee.

3.2.2 Marking

After packet classification, packet is marked with different Differentiated Service Code Point (DSCP) in DS field at header according to policies for further proceeding as Fig. 3-4 shows. IP header is 8-bits which contain precedence, Type Of Service (TOS) and Must Be Zero (MBZ). 3-bits are set to define packet precedence and 4-bits defines the TOS and last bit must be zero. After marking the header is replaced with 6-bits DSCP and last two bits are Current Unused (CU). Two DSCP which are Expedited Forwarding (EF) and Assured Forwarding (AF) are proposed in DiffServ.

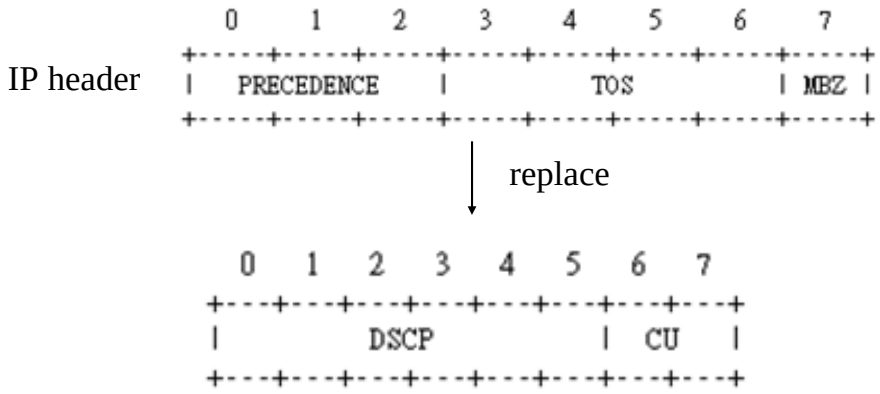

Fig. 3-4 DiffServ Code Point

3.2.3 Metering

Packets which are marked as EF will enter metering process for flow rate control avoiding flow rate of EF exceeds allocated bandwidth. Metering is implemented with token bucket which has two parameters, token generate rate and depth of bucket. If token is available then EF will be forwarded else drop EF group.

3.2.4 Forwarding

Packets are outputted with different PHBs and find router or protocol interface for forwarding. AF group are direct forwarding while EF are metering first then forwarding.

3.2.5 Shaping

Shaping works like metering but do not drop packets directly. Shaping buffering packets while token is not available instead. Packets are dropped while buffering is full. Shaping

is a optional function and may not be performed.

3.2.6 Per-Hop Behavior (PHB)

More detailed packet treatment in DiffServ routers is codified in the form of Per-Hop Behaviors (PHBs), which define the treatment of packets for "hops" between two routers that are adjacent to each other in the sense of IP routing topology. The currently standardized PHBs fall into two classes, the Expedited Forwarding (EF) PHB and the Assured Forwarding (AF) PHB class which shown in Table 3-1【14】【15】.

Table 3-1 AF and EF PHBs【14】【15】

<table>
<tr><td></td><td colspan="5">AF PHB</td><td>EF PHB</td><td>Best-effort</td></tr>
<tr><td>Features</td><td colspan="5">4 delay priority classes, each with 3 drop precedence subclasses</td><td>Premium/Virtual leased line service</td><td>none</td></tr>
<tr><td rowspan="4">Recommended DSCP</td><td>Delay⇨
Drop⇩</td><td>AF1</td><td>AF2</td><td>AF3</td><td>AF4</td><td rowspan="4">101110</td><td rowspan="4">000000</td></tr>
<tr><td>low</td><td>001010</td><td>010010</td><td>011010</td><td>100010</td></tr>
<tr><td>middle</td><td>001100</td><td>010100</td><td>011100</td><td>100100</td></tr>
<tr><td>high</td><td>001110</td><td>010110</td><td>011110</td><td>100110</td></tr>
<tr><td>Traffic Control</td><td colspan="5">Static SLA
Policing, Classification, Marking, RIO/WRED Scheduling</td><td>Dynamic SLA Policing, Classification, Marking, Priority/WFQ Scheduling</td><td>FIFO Scheduling</td></tr>
<tr><td>Non-Conforming Packets</td><td colspan="5">Re-mark as best-effort</td><td>Drop</td><td></td></tr>
<tr><td>RFC</td><td colspan="5">RFC 2597</td><td>RFC 2598</td><td></td></tr>
</table>

Differentiated Services Code Point (DSCP) is six bits in IP header field which is called

DS field to determine how packets are forwarded by nodes. AF PHB has four delay level and each level has three drop precedence. EF PHB is used for low latency, low delay and assured bandwidth services.

Although DiffServ are still being debated, edge based managers are needed to at least control the traffic within each subscribed last-mile access link (customer-side edges) or among many subscribed access links (ISP-side edges). The DiffServ provide architecture to differentiate service class and apply different treatment. It does not provide per-flow QoS but per-class QoS instead. DiffServ does not support service quality guarantee and negotiable quality control by itself and needs extra mechanisms to support them.

3.3 Connectivity-Service Service Model

Connectivity-Service service model is our reference service model in thesis. Connectivity/Service service model is high level model and could represent the future service model more exactly. Client-server service model which is most popular used today is a special case of this model. Client-server model is not suitable service model for the future service such peer-to-peer and VoIP.

Connectivity means the connectivity provider which is called network provider, too. Network provider provides the connectivity between end points such as users and service application servers. Service means the service provider which provide service such as video and Voice over IP (VoIP) service. Connectivity/Service service model allows SLAs between the service provider and network provider, between user and network provider and between user and service provider such as shown in Fig. 3-5. Connectivity/Service service model

provides dynamic QoS control, QoS guarantee and flexible QoS negotiation.

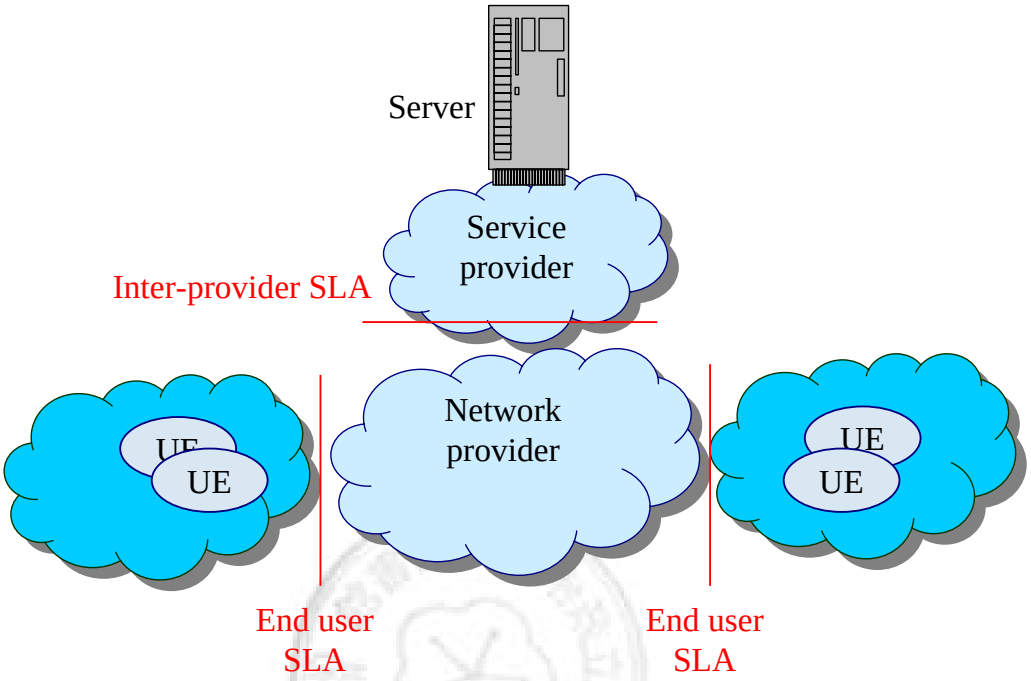

Fig. 3-5 Connectivity/Service Service Model

3.4 Mathematical Analysis

According to【2】, client-server based user modeling architecture provides six advantages listed blow.

- Information integrated in central repository.
- User Information can be reuse by more than one application at the same time.
- Information is stored in non-redundant manner. The consistency and coherence of information gathered by different application can be achieved more easily.
- Information can be maintained with low redundancy.

- System security, identification, authentication, access control and encryption can be applied for protecting user's information in server side.

So client-server architecture could help us modeling the user satisfaction. We place a QoS Agent (QA) on User Equipment (UE) to monitor application performance and report to QoS control center. Quality control center communicate with edge router with the quality feedback and edge router may change it's configure according quality feedback and current network status.

Users and application server are in the access network and connect to DiffServ Domain core network through edge router. For QoS control, we add Quality Control Center and Quality Agent on DiffServ as Fig. 3-6 shows. QoS Agent communicates with QoS Control Center to feedback QoS which user experienced. Then QoS Control Center can tell the edge router how to control network resource dynamically.

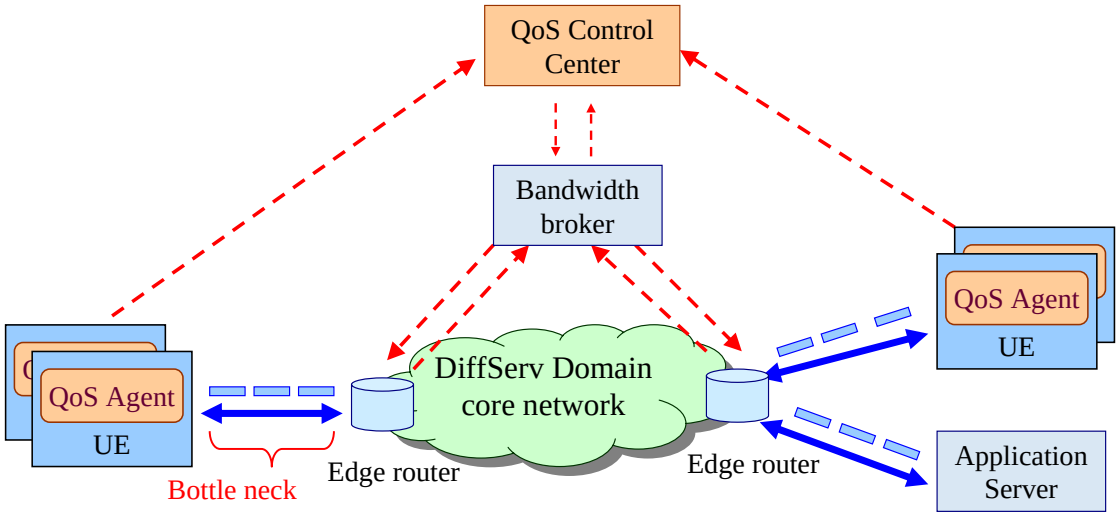

Fig. 3-6 Feedback-based QoS Control

3.4.1 QoS Feedback Algorithm

A QoS Agent (QA) is embedded in UE or implemented in application to monitor performance of application as Fig. 3-7 shows.

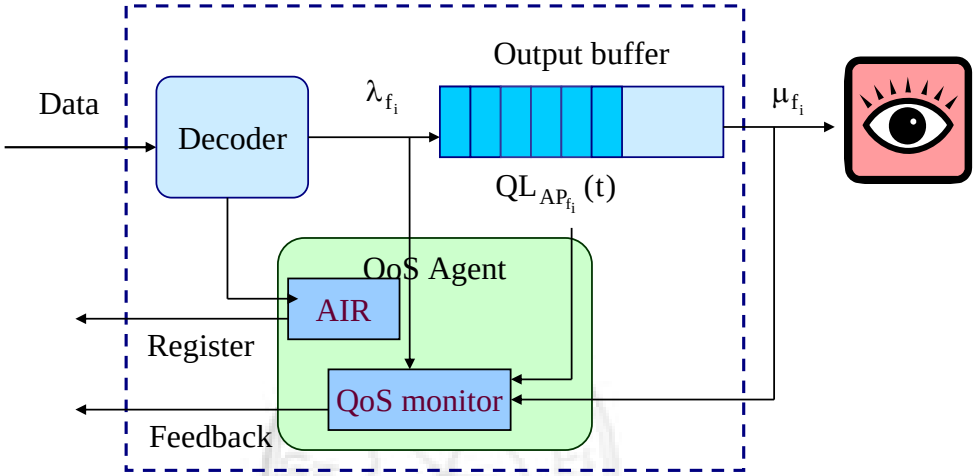

Fig. 3-7 QoS Agent

For example, QoS Agent monitors streaming video by snapshot frame rate and get the performance of application which list below. Streaming video is a special case which has more buffer length while in VoIP the buffer length may be shorter. Packets are arrival to receiver with packet arrival rate. After decoding in to frames and generate a frame rate such as Frame Per Second (FPS) λ_{f_i}. Frame rate could be modeled by traffic model or trace file. So number of frame $FN^{in}_{AP_{f_i}}(t)$ is a time function of frame rate. At a time instant TI_j we obtain a number of incoming frames such as Fig. 3-8 shows.

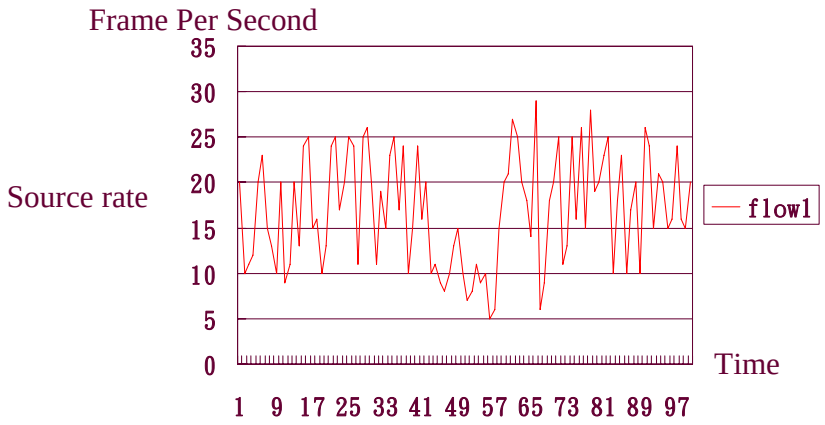

Fig. 3-8 Number of frame V.S. time

So the QoS of application AP is the throughput of application represented as number of frames and is calculated in following formula.

$$q_{f_i}^{delay}(TI_j) = FN_{AP_{f_i}}^{in}(TI_j) + QL_{AP_{f_i}}(TI_j) - FN_{AP_{f_i}}^{out}(TI_j) \tag{3-1}$$

$q_{f_i}^{delay}(TI_j)$ is the current throughput represented by frames. $FR_{AP_{f_i}}^{in}(TI_j)$ is the incoming frame rate of flow f_i which is calculated below.

$$FN_{AP_{f_i}}^{in}(TI_j) = \lambda_{f_i} * TI_j \tag{3-2}$$

λ_{f_i} is frame rate of flow f_i.

$QL_{AP_{f_i}}(TI_j)$ is the current queue length of flow f_i in output buffer at time TI_j. Queue length is number of frame in buffer. $FN_{AP_{f_i}}^{out}(TI_j)$ is current number of output frame at

time instant TI_j which is calculated below

$$FN_{AP_{f_i}}^{out}(TI_j) = \mu_{f_i} * TI_j \tag{3-3}$$

If current performance which here is the current frame rate for playing, less than required QoS, QoS Agent will signal a QoS alert. But the bad performance doesn't really represent the perception of user because the tolerance of user to application is different and the feeling of user to application is not directly equal to the application performance. While the performance of application drops for a shot period of time and recovered soon. User may feel nothing because the better tolerance of user or application to performance dropping. One user may be willing to pay more to get better performance because he could not tolerate bad one. Different service type of application also has different tolerance of performance. So we proposed a QoS Feedback Algorithm (QFA) achieving different tolerance of performance to closely represent user's satisfaction. The algorithm is shown in Fig. 3-9.

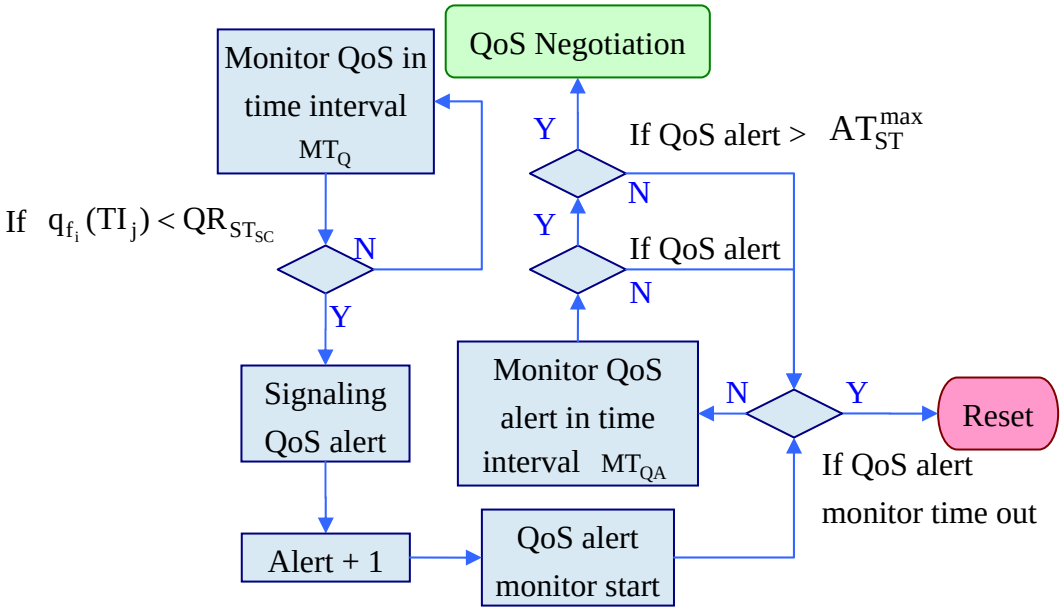

Fig. 3-9 QoS Feedback Algorithm

Current QoS of flow f_i at time instant TI_j is $q_{f_i}(TI_j)$. If the current QoS $q_{f_i}(TI_j)$ which is output frame rate dropped in QoS monitor time interval $MT_{Q_{ST_{SC}}}$, QoS Agent will find a bad QoS and signaling a QoS alert to QoS control center. QoS dropped means $q_{f_i}(TI_j)$ is less than the QoS requirement of service class SC which belong to service type ST. The QoS requirement is defined as $QR_{ST_{SC}}$. From the QoS feedback algorithm we can find one QoS decadence may doesn't make QoS control to start because user may feel nothing if just one bad QoS occurs. QoS control center counting the QoS alert times. The max QoS alert times which user can tolerant is different from service type. For example, video is a service type which could not tolerant too many QoS decadence and QoS alert. While QoS alert time exceed the maximum QoS alert time $AT_{ST_{SC}}^{max}$, QoS control center starting to perform QoS negotiation. If no more QoS alert to be find in a QoS alert time interval $MT_{QA_{ST_{SC}}}$.

QoS control center take it as QoS is acceptable and user doesn't feel bad QoS. Then QoS control center reset the QoS alert monitor status. The QoS feedback algorithm is based on bad QoS alert, so we can figure out that QoS is acceptable but couldn't represent the QoS is good.

3.4.2 QoS Tolerance

The times of QoS alert signaling which a user can tolerate implies the user tolerance to QoS. As Fig. 3-10 shows for example, QoS need not to be monitor every second because the user perception is not linear related to the performance of application. Monitor QoS and feedback in discrete monitor time interval could reduce loading and overhead of QoS feedback monitor. So QoS agent monitors the performance which is the throughput of application in a QoS monitor time interval. QoS monitor time interval is different from service types. The QoS monitor time interval for service type ST is defined as $MT_{Q_{ST}}$. While a QoS alert is signaled to QoS Control Center then QoS Control Center starting to count QoS alert in QoS alert monitor time interval. QoS alert monitor time interval is different from service types. The QoS alert monitor time interval of service type ST is defined as $MT_{QA_{ST}}$. The maximum QoS alert times AT_{ST}^{max} in Fig. 3-10 is 2 which means while QoS alert exceed 2 times in QoS alert monitor time interval $MT_{QA_{ST}}$. QoS control center take the QoS as too worse to tolerate for users. QoS tolerance should be defined by service provider.

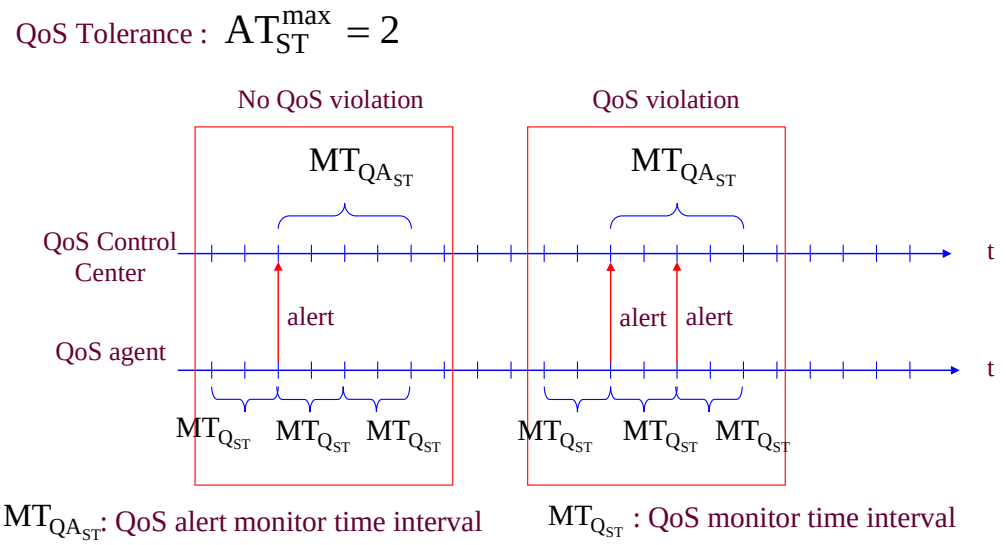

Fig. 3-10 QoS tolerance

So we can find the a maximum QoS alert times is the integer of ratio of QoS alert monitor time interval $MT_{QA_{ST}}$ and QoS monitor time interval $MT_{Q_{ST}}$ and QoS alert time of service type ST which is defined as AT_{ST} should be less or equal to the integer of ratio.

$$AT_{ST} \le \left\lfloor \frac{MT_{QA_{ST}}}{MT_{Q_{ST}}} \right\rfloor, MT_{QA_{ST}} \ge MT_{Q_{ST}} \qquad (3\text{-}4)$$

3.4.3 QoS Sensitivity

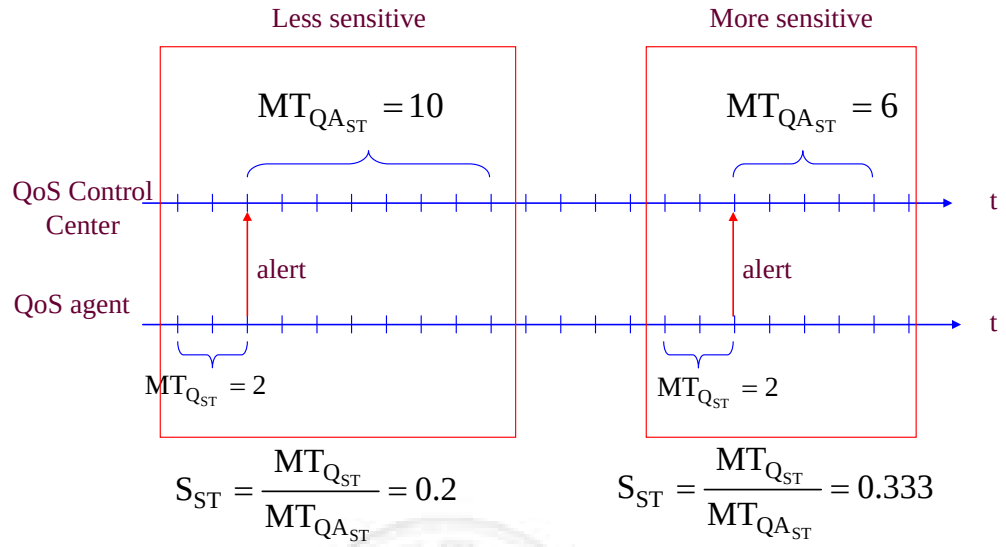

Fig. 3-11 QoS sensitivity

As Fig. 3-11 shows, QoS Agent monitors QoS in QoS monitor time interval $MT_{QA_{ST}}$. The shorter $MT_{Q_{ST}}$ the higher possibility of bad QoS would be found. So QoS sensitivity is inverse proportion to QT. QoS control center monitor QoS alert in QoS alert monitor time interval $MT_{QA_{ST}}$. Longer $MT_{QA_{ST}}$ make the response of bad QoS slower which implies the sensitivity to QoS is lower. Different service type ST has different sensitivity. Sensitivity of service type ST is defined as S_{ST}. We can find the relation of QoS sensitivity S_{ST}, $MT_{Q_{ST}}$ and $MT_{QA_{ST}}$ is listed below :

$$S_{ST} = \frac{MT_{Q_{ST}}}{MT_{QA_{ST}}}, MT_{Q_{ST}} \le MT_{QA_{ST}} \quad (3\text{-}5)$$

$MT_{Q_{ST}}$ must be less or equal to MT_{ST}^{QA} so at last one QoS alert can be seen. From formula (3-1) and (3-4) we obtain a result which QoS tolerance is inverse proportional to QoS sensitivity listed in (3-7).

$$AT_{ST} \le \left\lfloor \frac{MT_{QA_{ST}}}{MT_{Q_{ST}}} \right\rfloor = \left\lfloor \frac{1}{S_{ST}} \right\rfloor, MT_{QA_{ST}} \ge MT_{Q_{ST}} \tag{3-6}$$

$\lfloor\ \rfloor$ is gauss symbol and we obtaining the lower bound of gauss as maximum QoS alert time value. And the maximum of the accepted alert times of service type ST which is defined as AT_{ST}^{max} is:

$$AT_{ST}^{max} = \left\lfloor \frac{1}{S_{ST}} \right\rfloor \tag{3-7}$$

3.4.4 User Perception of Delay

Delay is the throughput of applications too low such as Frames Per Second (FPS) lower than the requirement of user. Delay of application which generate traffic flow i is defined as $q_{f_i}^{delay}$. From formula (3-1), calculation of delay is listed below.

$$q_{f_i}^{delay} = FN_{AP_{f_i}}^{in}(TI_j) + QL_{AP_{f_i}}(TI_j) - FN_{AP_{f_i}}^{out}(TI_j) \tag{3-8}$$

Where $FN_{AP_{f_i}}^{in}(TI_j)$ is the number of incoming frame of flow f_i at time instant TI_j.

Frame rate of flow f_i in buffer at time instant TI_j is $QL_{AP_{f_i}}(TI_j)$. Number of output frame of flow f_i at time instant TI_j is $FN^{out}_{AP_{f_i}}(TI_j)$.

If $q^{delay}_{f_i}$ less than the minimum QoS requirement QR^{dealy}_{ST} then user may feel delay. $q^{delay}_{f_i}$ is represented by number of frame at time instant TI_j. Using the QoS monitor algorithm to monitor delay. Different delay sensitivity could be achieved by adjusting QoS monitor time interval and QoS alert monitor time interval.

3.4.5 Queuing Model

Service classes in DiffServ could be implemented with queues. The most popular queuing technique is Class-Based Queuing (CBQ).

3.4.5.1 Class-Based Queuing (CBQ)

CBQ come from the idea of link-sharing【8】. The requirements for link-sharing are essentially the same whether the link-sharing is between service classes, organizations, protocol families, or traffic types. Link-sharing means requirements from different agency share one link. While network was congested, each agency still has guaranteed bandwidth allocated to them and allows lending unused bandwidth to other agency. This promotes the utilization of network. Packets in the queues are waiting to be scheduled out according to their specific parameters. Many packet schedulers have been proposed. The most popular one is Class-Base Queuing (CBQ) approach. Using Per-class classifier to classify packet

belonging to different service classes such as shown in Fig. 3-12.

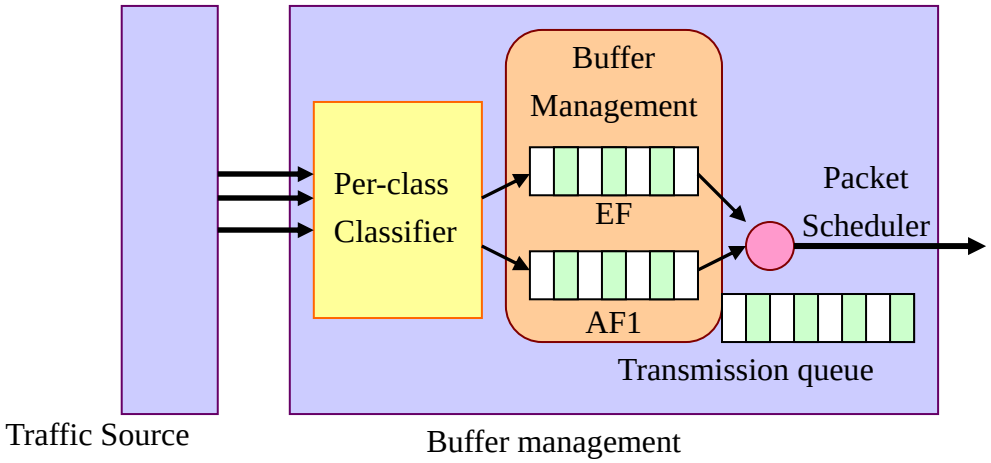

Fig. 3-12 Resource Manager Architecture

One requirement for link-sharing is to share bandwidth on a link between multiple organizations, where each organization wants to receive a guaranteed share of the link bandwidth during congestion, but where bandwidth that is not being used by one organization should be available to other organizations sharing the link. Bandwidth can be shared by different agencies, protocols and traffic types such as Fig. 3-13 shows.

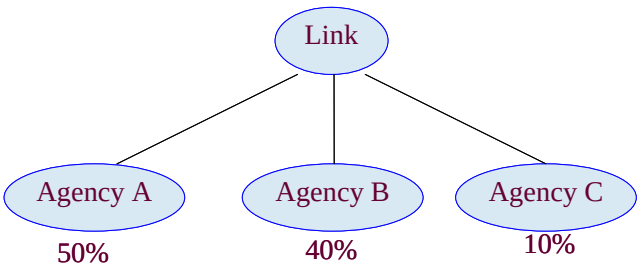

Fig. 3-13 Link-sharing between multiple agencies or protocols

3.4.5.2 Hierarchical CBQ (HCBQ)

Original CBQ mentioned above is like flat structure. Hierarchical support hierarchical link sharing to provide more resilience usage of bandwidth. Fig. 3-14 shows the hierarchical class-based bandwidth allocation. Hierarchical link-sharing is more powerful and flexible on bandwidth usage and borrowing than flat CBQ.

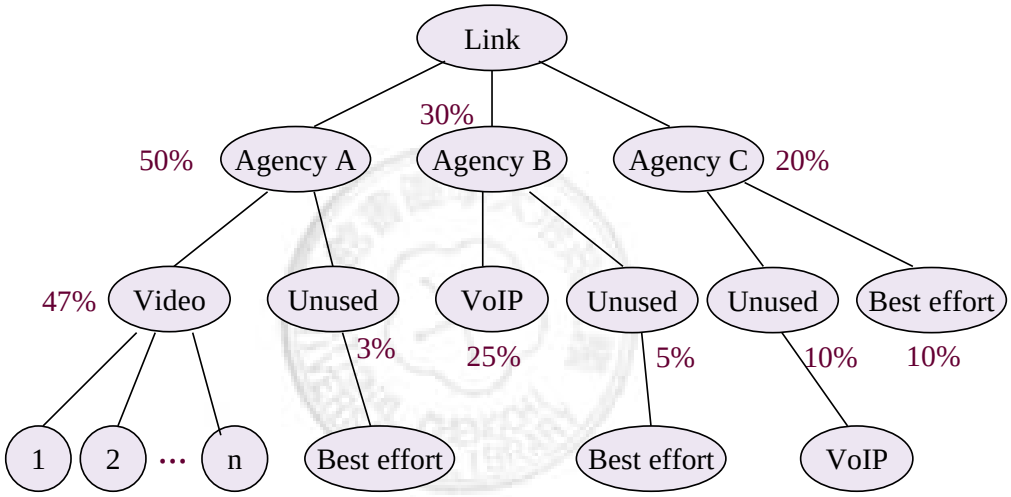

Fig. 3-14 A hierarchical link-sharing structure

Bandwidth is allocated to each service class of DiffServ with different volume as shown in Fig. 3-15 and could share bandwidth with each other. So the sum of percentage of bandwidth allocation to different service classes may not be equal to one because of bandwidth borrowing.

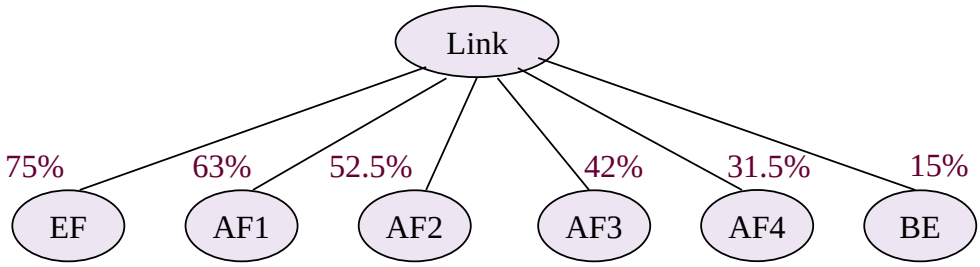

Fig. 3-15 Bandwidth sharing in DiffServ

However, two problems arise due to multiple competing TCP sessions within each CBQ class【4】:

- Large buffer requirement. A TCP session with a large TCP receiver window (RWND), if bounded by a very limited bandwidth such as bottleneck which bandwidth manager creates, may cause a huge number of packets be queued at the bottleneck. So the bandwidth manager should prepare a large memory to store the queued packets.
- Fairness among sessions within the same class. Given a small congestion window, a TCP session with a large Round Trip Time (RTT) will obtain a small bandwidth.

3.4.5.3 Per-Flow Queuing (PFQ)

In different class queues, packets form different sources of many sessions can be seen as flows. Flows in the same service class queue will scramble resource with each others which cause fairness problem, so flow isolation and management is desired. Function of Per-Flow Queuing (PFQ) as Fig. 3-16 shows.

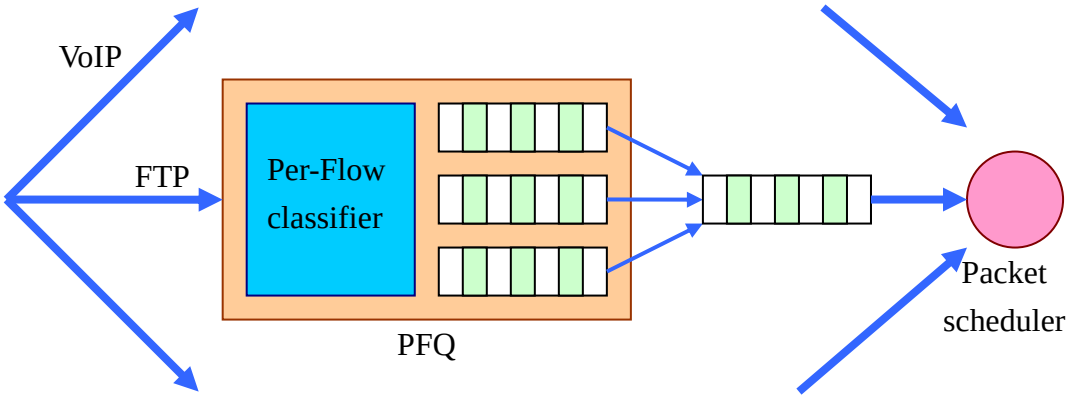

Fig. 3-16 Per-Flow Queuing

3.4.5.4 Flow Control

DiffServ are implemented with a set of queues. Each queue is associated with a specific service class. A DSCP is assigned to a service class and a PHB is mapped to actions on queues. Assuming the classification of packets to different service classes is already defined. We focus on the negotiation with packet classifier using feedback from end user to dynamic change the service mapping.

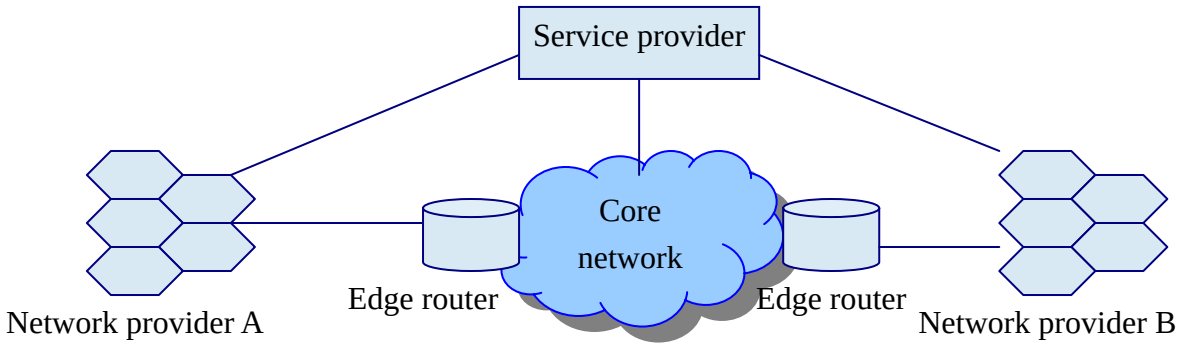

Fig. 3-17 Network and service provider model

User may sign up with different network provider and use different access network. Network resource for one service will different in different network provider as Fig. 3-17 shows.

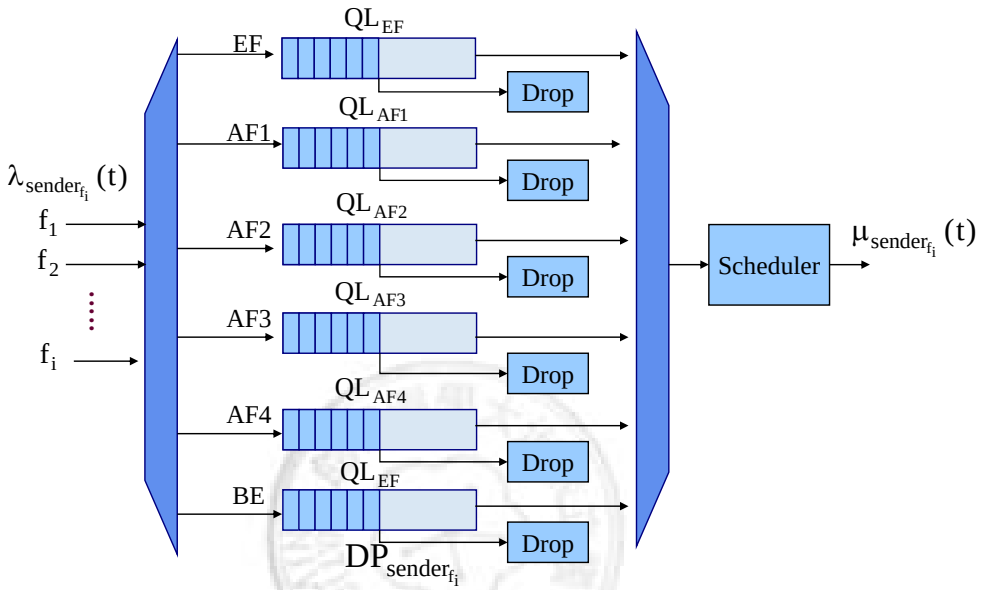

Fig. 3-18 Per-class queuing

Queuing model in sender side for example, is shown in Fig. 3-18. Flows enter sender router are classified to DiffServ service classes and one Best Effort (BE) class by default service mapping. The DiffServ service classes are Expected Forwarding (EF), Assured Forwarding (AF) 1, AF 2, AF 3 and AF 4.

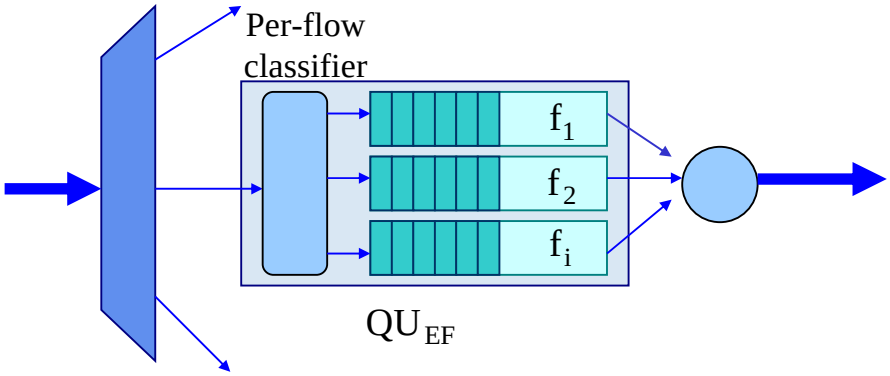

Fig. 3-19 Per-flow control in DiffServ

Several flows are classified into the same class will scramble resource allocated to this class which cause throughput descend as Fig. 3-19 shows. Each service class is managed with queue which queue length is QL. For example, the queue length of EF at sender side is QL^{EF}_{sender} and could be represented as the number of frames. Current queue length at time t is a time function to number of enqueued frames and is defined as a rate $QL(t)$. Packets of flow f_i after classifying and marking packet's DSCP field are placed in appropriate queue. Each queue QU, has an associated weight as shown in Fig. 3-19. The weight of EF queue at sender side is $w_{QU^{EF}_{sender}}$. If the queue size exceeds the maximum agreed length, packets will be dropped. The dropping probability of a packet belong to flow f_i in queue at sender side QU^{EF}_{sender} is :

$$DP_{sender_{f_i}} = \frac{w_{QU^{EF}_{sender}} + w_{sender_{f_i}}}{TBW_{sender_{f_i}} * NT_{f_i}} * \bar{q}_{f_i} \qquad (3\text{-}9)$$

Where $w_{sender_{f_i}}$ is the weight of flow f_i at sender side which the packet belongs to. $TBW_{sender_{f_i}}$ is target bandwidth of flow f_i at sender side. By the setting of target

bandwidth could guarantee the minimum per-flow requirement and prevent flow from starvation. Target bandwidth could be calculated by expected Media Unit throughput which we take as QoS requirement $QR_{ST_{SC}}$ and media unit size MUZ_{ST}.

$$TBW_{f_i} = QR_{ST_{SC}} * MUZ_{ST} \tag{3-10}$$

Where TBW_{f_i} is target bandwidth of flow f_i which belong to service type ST. Because the user could sign up with different network provider so the target bandwidth may be different at sender side and receiver side and are defined as $TBW_{sender_{f_i}}$ and $TBW_{receiver_{f_i}}$. We could assume them are the same in default.

$\overline{q}_{f_i}$ is the average QoS of flow f_i represented by frame rate. NT_{f_i} is QoS negation times of flow f_i. More times of QoS negotiation means the QoS of the flow f_i is often bad. So we should increase the packets throughput of the flow f_i. We could decrease the drop probability of this flow f_i while the QoS negotiation times increased and implies the drop probability is inverse proportional to QoS negotiation times. The higher bandwidth requirement implies the lower dropping probability. The higher QoS implies the higher dropping probability. $\overline{q}_{f_i}$ is average frame rate during monitor time interval MT_{QA} calculated in:

$$\overline{q}_{f_i} = \frac{\sum_{j=1}^{AT_{ST}^{max}} q_{f_i}(TI_j)}{MT_{QA}} \tag{3-11}$$

Decadence in QoS of flow f_i means the packet throughput of flow f_i too low. We could increase the throughput by lower the dropping probability of flow f_i. The lower the $\overline{q}_{f_i}$, the lower the drop probability. By the building of feedback-based QoS control framework, flow control will be more intelligent.

The drop probability $DP_{sender_{f_i}}$ of a packet in queue QU_{AF1} at sender side is

$$DP_{sender_{f_i}} = \frac{w_{QU^{AF1}_{sender}} + w_{sender_{f_i}}}{TBW_{sender_{f_i}} * NT_{f_i}} * \overline{q}_{f_i} \quad (3\text{-}12)$$

And so on, we can obtain drop probability of flow f_i in each queue at sender side. The drop probability of flow f_i in each queue at receiver side is analogized in $DP_{receiver_{f_i}}$

$$DP_{receiver_{f_i}} = \frac{w_{QU^{EF}_{receiver}} + w_{receiver_{f_i}}}{TBW_{receiver_{f_i}} * NT_{f_i}} * \overline{q}_{f_i} \quad (3\text{-}13)$$

3.4.6 Network Model

We use the network model as shown in Fig. 3-20. Flow f_i is generated by one user at sender network and is sent through router called sender router. The traffic generated by flow f_i is sent to sender router and could be seen as arrival rate of flow f_i from the point of sender router. The arrival rate at time instant TI_j is defined as $\lambda_{sender_{f_i}}(TI_j)$. In sender router flows are controlled and then output to core network formed a output rate. The output

rate at time instant TI_j is defined as $\mu_{sender_{f_i}}(TI_j)$. After traveling through core network, packets arrived in router which belongs to receiver and defined as receiver router. The incoming rate can be seen as arrival rate for receiver router. The arrival rate of flow f_i at time instant TI_j is defined as $\lambda_{receiver_{f_i}}(TI_j)$. Flows are controlled at receiver router and are sent to user in a output rate. The output rate of flow f_i at time instant TI_j is defined as $\mu_{receiver_{f_i}}(TI_j)$.

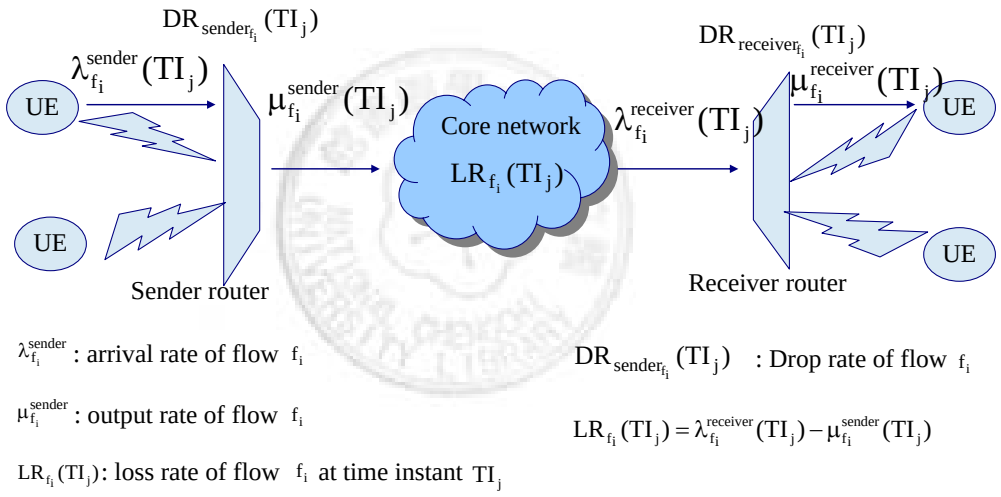

Fig. 3-20 Network Model

UE moves from one access network to target access network will increasing the loading of target access network and cause burst traffic at edge router connect to target access network. Many packets may be dropped due to congestion and resulting in high packet loss rate. Traffic from heterogeneous network cause the congestion of edge router and many packets may be dropped resulting in high packet loss rate. We narrow down the problem by assuming we have a well power control mechanism and stable assess network. The packet

loss due to power propagation or other reason in wireless network is assumed to be well controlled. The packet loss rate in access network such as sender side due to some power or radio error at time instant TI_j is defined as $LR^{error}_{sender_{f_i}}(TI_j)$.

3.4.7 QoS Negotiation

When QoS Negotiation is requested, QoS Control Center should know the situation of network and try to find what problems may cause QoS drop and try to negotiate with network elements. Negotiation means the goal may not be achieved or the problem may not be solve even the problem may not very clear but could obtain a results may be better or satisfied most of people. Here are three main parts need to be described for QoS negotiation.

3.4.7.1 Part 1: Sender Side Access Network

Arrival rate of flow f_i to sender router may be different from its output rate from sender router because of flow control. While sender access network is under the situation of high loading, the difference between arrival rate and output rate is larger and delay of flow f_i may occurs. In this situation, weight of flow f_i which is defined as w_{f_i} should be increased.

The status of sender side access network could be figure out by the throughput of flow f_i which is defined as $TP_{sender_{f_i}}$. Because flow f_i is come from access network with error rate $LR^{error}_{sender_{f_i}}$. So the packets succeed sent to sender router is $\lambda_{sender_{f_i}} * (1 - LR^{error}_{sender_{f_i}})$. Some packets are queued in queue at sender router with drop probability $DP_{sender_{f_i}}$. The packets in

queue will be sent in the probability of $(1-DP_{sender_{f_i}})$. So the throughput of flow f_i at sender side $TP_{sender_{f_i}}$ is:

$$TP_{sender_{f_i}} = \mu_{sender_{f_i}} - \lambda_{sender_{f_i}} * (1 - LR^{error}_{sender_{f_i}}) + \frac{QL_{sender} * (1 - DP_{sender_{f_i}})}{t} \qquad (3\text{-}14)$$

While $TP_{sender_{f_i}}$ below a threshold value which is defined as $TH_{sender_{f_i}}$ implies that congestion of flow f_i occurs at sender side.

3.4.7.2 Part 2: Receiver Side Access Network

This situation is like the situation in sender side. But the error in receiver access network may occur while packets of flow f_i are sent through access network. The flow rate of flow f_i from receiver router to user through access network could be seem as output rate of receiver router and is defined as $\mu_{receiver_{f_i}}$. So the receiver may encounter access error in access network with the error rate $LR^{error}_{receiver_{f_i}}$ and the packets successfully received by receiver is $\mu_{receiver_{f_i}} * (1 - LR^{error}_{receiver_{f_i}})$. The throughput of receiver side access network is $TP_{receiver_{f_i}}$ which is defined below:

$$TP_{receiver_{f_i}} = \mu_{receiver_{f_i}} * (1 - LR^{error}_{receiver_{f_i}}) - \lambda_{receiver_{f_i}} + \frac{QL_{receiver}(t) * (1 - DP_{receiver_{f_i}})}{t} \qquad (3\text{-}15)$$

While the threshold of flow f_i at receiver side is $TH_{receiver_{f_i}}$.

3.4.7.3 Part 3: Core Network

If the sender side and receiver side are all in normal state which means throughput of flow f_i is not delayed and dropped too over. The problem may occur in core network. Complicated and hardly be controlled in core network. In DiffServ we could change the DSCP of packets belonged to flow f_i at edge router, and then packets could be treated better and forwarded quickly while travel through core network.

We simplified the packet loss rate by watching the input and output rate at edges of core network and calculating the packet loss rate by comparing input and output rate. Impact may caused by different distribution of packet loss rate but we assume the impact is only the variance of packet loss rate and not very import to our research. Various routing techniques and other management in core network will effect the distribution of packet loss rate. We may apply different distribution to packet loss rate in order to represent the situation of core network more exactly. In this thesis we use the simplest packet loss rate to reduce the influence of other control management and emphasize the effect of our feedback-based QoS control.

The packet loss rate of flow f_i at time instant TI_j while travel in core network which is defined as $LR_{f_i}(TI_j)$ could be calculated in formula (3-16)

$$LR_{f_i}(TI_j) = \lambda_{receiver_{f_i}}(TI_j) - \mu_{sender_{f_i}}(TI_j) \quad (3\text{-}16)$$

3.4.7.4 QoS Negotiation Algorithm

We can dynamic changing weight of flow by application feedback $\bar{q}_{f_i}$ and negotiation times NT_{f_i}. The weight is also automatically changed according to the feedback of application. We don't want to change the entire network at one time but control part by part. If sender side has problem we control flows in sender side. This reduces the impact to other flows and network for the entire networks. DSCP affects the transmission behavior of packets in network. If the problem of bad QoS is network, we can control QoS by changing DSCP and send to network and the transmission efficiency of packets may get recovery. We propose a QoS Negotiation Algorithm to handle the situations and negotiate QoS as Fig. 3-21 shows.

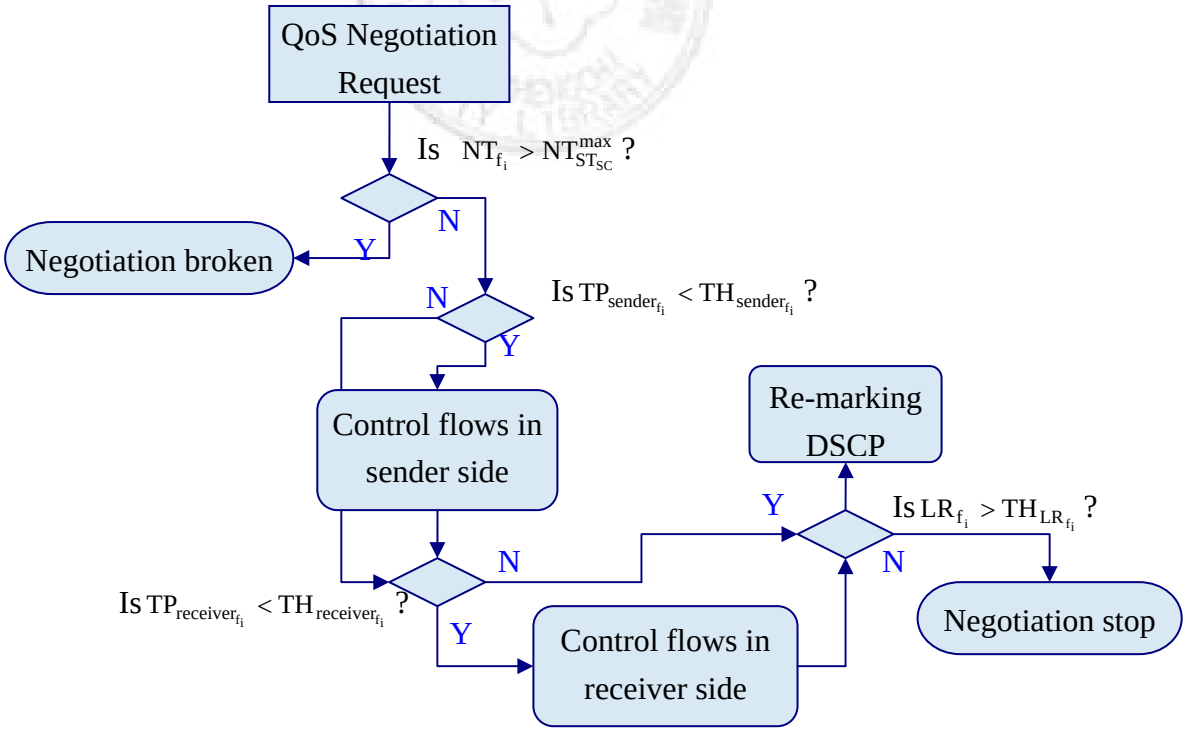

Fig. 3-21 QoS Negotiation Algorithm

3.4.7.5 Part 4: Negotiation Broken

If QoS is negotiated in all the situations but user still receive bad QoS and negotiation exceeds the maximum negotiation times. The QoS negotiation is broken. This show the network may have unrecoverable failure. The session may be needed to be terminated because QoS is to bad to be tolerant. Disconnect directly is better than suffering bad QoS continually or be dropped after waiting too long for expected QoS recovering from the perspective of user. User can try to initiate a new session to reset the resource allocation if network support the ability. This situation could be caused by user may be even doesn't have enough right to enjoy the good QoS. So while network congestion they may receive slower QoS recovery and bad QoS. User could pay more for higher QoS.

3.5 Enhanced Service Level Agreement (SLA)

Service Level Agreement (SLA) is used for documenting the level of service between the customer and a service provider or network provider as Fig. 3-22 shows. To provide per-flow QoS guarantee, user QoS requirements need to be specified in SLA. The SLA between end user and service provider is necessary. In DiffServ SLA contains Traffic Condition Agreement (TCA) which defines the traffic characteristic and not enough for QoS guarantee. For per flow QoS guarantee, we need to describe QoS agreement and QoS specification.

User performs service provided by service provider through network connectivity which is provided by network provider. Service could be differentiated in to different service class with different QoS provided. User pays more for better QoS and need the support of network

provider because high QoS may need high throughput of network. Service provider wants to provide services with different QoS need to make inter-provider SLA with network provider for QoS mapping to network. So the SLA between service provider and end user should be integrated with SLA between network provider and end user.

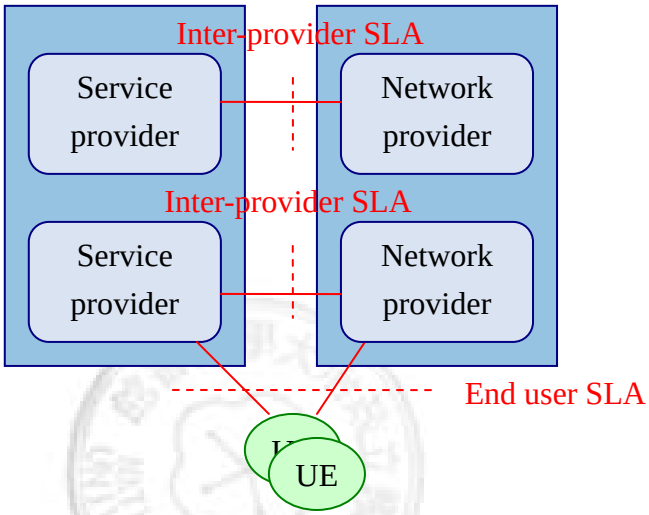

Fig. 3-22 SLAs Model

User can subscribe with different network provider which should provide basic connectivity such as best effort. Inter-provider SLA enhances the QoS guarantee. One question is risen when service provider provide different service class but doesn't subscribes inter-provider SLA with network provider. User may subscribe a platinum service but network provider which user subscribed with doesn't provide suitable network resource because service provider doesn't subscribe inter-provider SLA with this network provider and network provider provides only best effort for the service provider. For prevent this situation, a third party authentically organization or elements called Service Level Agreement Management is needed to handle the service QoS and network connectivity.

3.5.1 Inter-Provider SLA

Inter-provider SLA is contract described network management such as classify rule and network resource requirement such as bandwidth, throughput. In this thesis we extend the inter-provider SLA which listed in Table 3-2.

Table 3-2 Inter-Provider SLA

Field	Parameter	Unit/meaning
Service Type	ST	Video, VoIP, ...
Service class	SC	Platinum, gold, ...
Classify Rule	CR	EF, AF1, ...
Target bandwidth	$TBW_{ST_{SC}}$	bps
Min. throughput	TP_{ST}^{min}	bps
QoS monitor time interval	$MT_{QST_{SC}}$	second
QoS alert monitor time interval	MT_{QA}	second
Max. QoS alert times	$AT_{ST_{SC}}^{max}$	times
Max. QoS negotiation times	$NT_{ST_{SC}}^{max}$	times

Service type could have multiple services classed such as platinum with highest video quality. Service are classified to specified DiffServ class by classify rule. Target bandwidth for service class SC of service type ST which is defined as $TBW_{ST_{SC}}$ is the target bandwidth of the service provider wish network provider to provide for this service. For QoS guarantee, the value should be the minimum value that service can tolerant. Min. QoS requirement is the minimum QoS requirement of user. For example, the min. QoS requirement of video is the frame rate (FPS) should keep at last 15 FPS or above. If the application encounters bad performance such as low FPS, QoS Agent will signal a QoS alert AT_{ST}. The maximum times of bad application performance can user tolerance is the Max. QoS alert time AT_{ST}^{max}. If QoS alert exceed the Max. QoS alert time, QoS control center will

perform QoS negotiation. For the sake of reducing system loading and avoid dead lock of system caused by un-coverable problem such as network failure. There should be upper bound of QoS negotiation times. QoS negotiation times could also be used to adjust the weight of the flow f_i.

3.5.2 End User SLA

The central part of end user SLA is the charging model between the end user and the network provider. Volume-based and time-based charging or combinations are used in the contract. In order to guarantee QoS and describing QoS requirement of end user, following specification of end user SLA is discussed.

End user SLA could be simplified as Table 3-3 shows, because the QoS requirement and network resource specification are already signed in inter-provider SLA. Service provider needn't signs trivial setting of parameters for QoS guarantee with user but provide abstract service classes instead such as platinum, gold, silver class. Providing set of service classes for user to select, such as high quality video in platinum service class which means frames rate always above 25 FPS and sensitivity is higher than lower level of service. Higher sensitivity results in faster QoS recovery when network is in high loading situation.

Table 3-3 End user SLA

Field	Parameter	Unit/meaning
Service Type	ST	Video, VoIP, ...
Service class	SC	Platinum, gold, ...

3.6 Feedback-Based QoS Control Framework

Future network architecture such as UMTS is divided in to control layer and transport layer for management resilience and convenience. In the layered architecture, control centers such as bandwidth broker and session control function are proposed. So our feedback-based QoS control could be easily integrated and implemented in future network architecture such as UMTS.

We propose a Feedback-based QoS control framework shown in Fig. 3-23. For the convenient realize for service provider and monitor QoS of end user, QoS Agent is proposed to provide a interface for service provider to be benefited from the advantage of UMTS and prevent complete procedure of setting the application. The performance of application could be monitored and transform into user's perception.

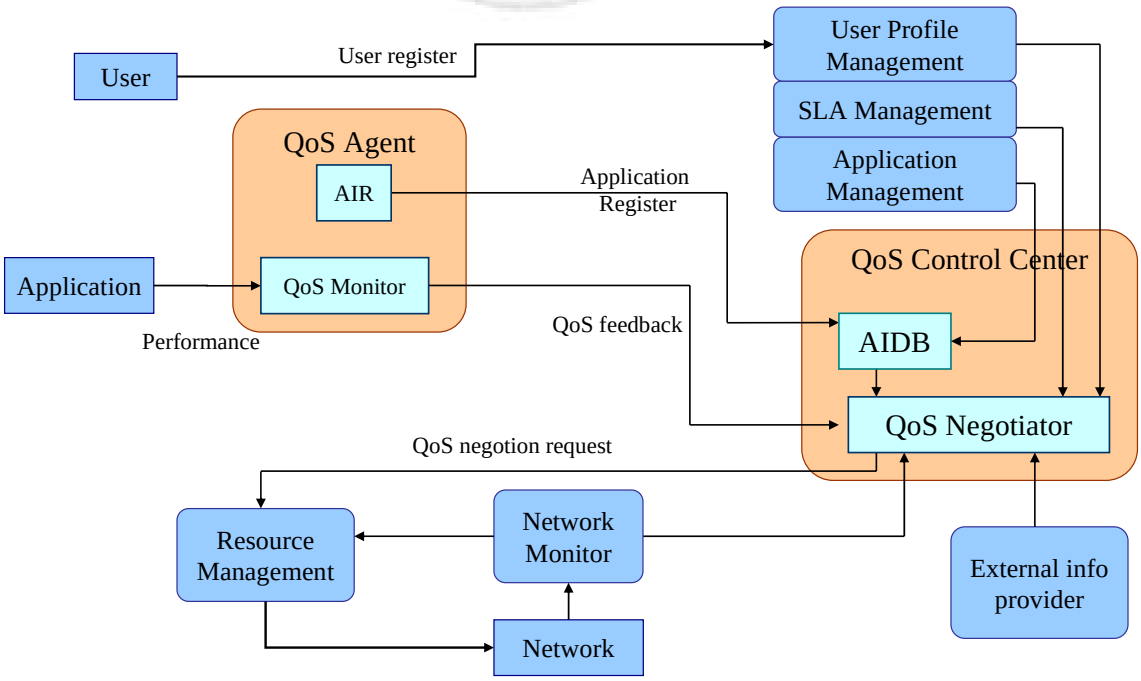

Fig. 3-23 Feedback-based QoS control Framework

3.6.1 QoS Agent (QA)

For downward compatibility of the applications, QoS control interface could be implemented as middleware such as Quality Agent. Quality Agent can coexist with the applications to monitor application's performance. While the performance is lower than the threshold of the desired QoS requirement, QA signals a QoS alert. Quality Agent could be placed as Fig. 3-7 shows.

QA is embedded on User Equipment (UE) or is installed or implemented in application to monitor the performance of application. Using QoS Feedback Algorithm (QFA) to transform the performance of application to user's perception and signaling a response to QoS control center while encounter bad QoS.

3.6.1.1 QoS Monitor (QM)

QoS Monitor monitors the performance of application and translating it to user's perception for feedback.

3.6.1.2 Application Instance Register (AIR)

Applications are registered in Application Instance Database (AIDB) through Application Instance Register (AIR). AIDB records the Traffic Specification (TSpec), Requirement Specification (RSpec) and QoS Specification (QSpec) of the application. TSpec describes the character of traffic such as packet size, peak rate, etc. RSpec describes the expected quality and defines the requirement for reservation. QSpec carries the information on resources

available, resource required, traffic descriptions and other information required by the Resource Management Function (RMF).

3.6.2 User Profile Management

User profile management manages user's profile which contains the Service Level Agreements.

3.6.3 SLA Management

SLA management manages the SLAs between providers and between provider and users. SLA management prevents SLAs from mismatching.

3.6.4 Application Management

Application management contains profile of application and QoS specification of different type of service.

3.6.5 Application Instance Database (AIDB)

Application Instance Database (AIDB) record the application instance registered through QoS Agent for monitor QoS feedback and QoS control.

3.6.6 QoS Negotiator

QoS negotiation negotiates with network resource management. Using QoS requirement, network status and user profile as input to negotiate.

3.6.7 External Info Provider

External Info Provider provides other information such as equipment provider may provide specification of equipment. Service provider could use equipment information to provide performance adjustment for specific equipment.

3.6.8 Network Monitor

Network Monitor reports the network status. Figure out the current network situation for QoS negotiation.

3.6.9 Resource Management

Resource management manages network resource such as bandwidth, flow rate and throughput. Bandwidth broker is the main function of resource management in DiffServ. By using QoS feedback we could enhance the resource management to be more intelligent.

3.6.10 Interfaces of Feedback-Based QoS Control

Logical functions of Feedback-based QoS control is shown in Fig. 3-24. Interfaces are need

for using QoS feedback. Interfaces could be plug-ins embedded in equipment and software for existent application or be implemented in new applications.

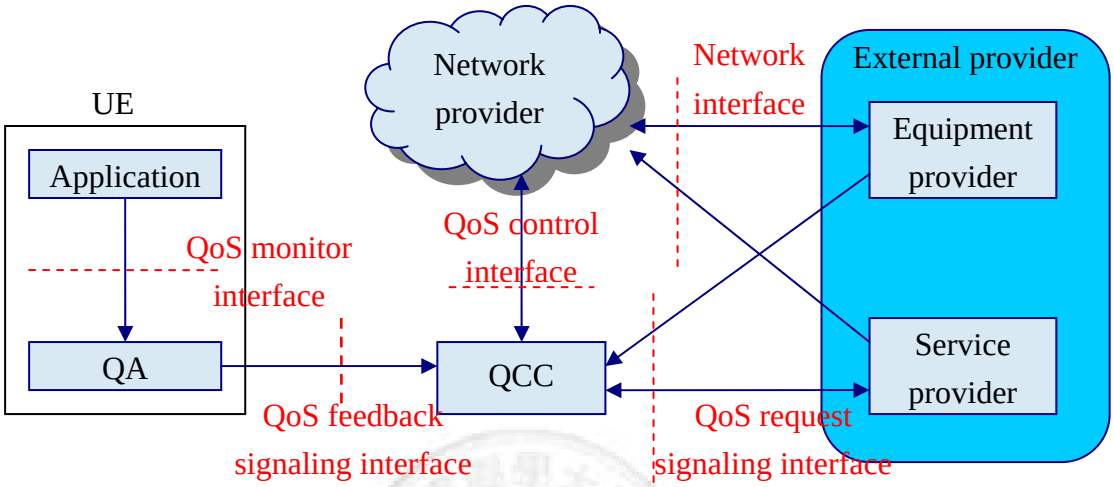

Fig. 3-24 QoS control interfaces

3.6.10.1 QoS Monitor Interface

QoS monitor interface is a communicating bridge between application and QoS Agent. According to 【18】, to guarantee QoS we need specify the QoS requirement which could be defined in QoS Specification (QSpec). Applications could use these parameters provide by QoS Specification (QSpec) to signal QoS feedback. QoS agent will register application and obtain a application ID (AID) and knowing how to monitor this application for QoS feedback. One Application generates a traffic flow and we defile the i-th flow as f_i. Media unit of specific service type is defined as MU_{ST} such as frame is media unit of video. Video data arrival to user through network and is decoded to frames. Incoming media unit of video which forms a flow f_i at time instant TI_j is $FN^{in}_{AP_{f_i}}(TI_j)$ where the suffix AP means

application for distinguishes the rate in network. Frames are buffered in application buffer after decoding. The current number of frames in buffer at time instant is defined as $QL_{AP_{f_i}}(TI_j)$. Frames in buffer are output while playing and is defined as $FN^{out}_{AP_{f_i}}(TI_j)$. QoS agent monitor QoS of application in the time interval $MT_{Q_{ST_{SC}}}$. While the QoS below the QoS requirement $QR_{ST_{SC}}$, QoS agent singling a QoS alert.

Table 3-4 QoS monitor interface

Field	Parameter	Unit/meaning
Application ID	AID	number
Media Unit	MU_{ST}	frame
Number of incoming Media Unit	$FN^{in}_{AP_{f_i}}(TI_j)$	frames
Number of output Media Unit	$FN^{out}_{AP_{f_i}}(TI_j)$	frames
Media Unit in buffer at time TI_j	$QL_{AP_{f_i}}(TI_j)$	frames
QoS monitor time interval	$MT_{Q_{ST_{SC}}}$	second
Min. QoS requirement	$QR_{ST_{SC}}$	frames/second

3.6.10.2 QoS Feedback Interface

Applications using by user will be registered by this interface and specify related parameters for specific application. QoS is feed backed by the descriptions through the interface to tell QoS control center how the satisfactions of user. The interface parameters are shown in Table 3-5. A new application is performed QoS agent will register this application through Application Instance Register (AIR) and obtain a application ID. QoS control center know what QoS requirement of the application is need and tell QoS agent how to monitor and feedback QoS. QoS control center identifies application by AID and identify flows by flow

ID f_i. If QoS of application below the QoS requirement, QoS agent will signaling a QoS alert and feedback the current QoS $q_{f_i}(TI_j)$ at time instant TI_j.

Table 3-5 QoS feedback interface

Field	Parameter	Unit/meaning
Service Type	ST	Video, VoIP, ...
Application ID	AID	Used for application registration
Flow ID	f_i	Number
Current QoS	$q_{f_i}(TI_j)$	Frames

3.6.10.3 QoS Control Interface

QoS control interface is provided for network provider, service provider and equipment provider for QoS specification. After QoS alert signaling, QoS control center will starting monitor QoS alert in QoS alert monitor interval $MT_{QA_{ST_{SC}}}$ and calculating average QoS $\bar{q}_{f_i}$. If QoS alert times exceeds the maximum QoS alert times $AT_{ST_{SC}}^{max}$, QoS control center will communicate with resource management with average QoS $\bar{q}_{f_i}$ and QoS negotiation times NT_{f_i} to negotiate QoS.

Table 3-6 QoS control interface

Field	Parameter	Unit/meaning
Service Type	ST	Video, VoIP, ...
Service Class	SC	Platinum, gold, ...
QoS alert monitor time interval	$MT_{QA_{ST_{SC}}}$	Second

Average QoS	$\overline{q}_{f_i}$	Frames/second
QoS Negotiation times	NT_{f_i}	Times
Min. QoS requirement	$QR_{ST_{SC}}$	Frames/second

3.6.10.4 QoS Requirement Interface

QoS requirements of service from service provider are communicated through QoS requirement interface which is shown in Table 3-7. QoS requirement of service type for different service classes is defined as $QR_{ST_{SC}}$ such as QoS requirement of platinum video service class is 25 FPS. Maximum QoS alert time of service type for specific service class is defined as $AT_{ST_{SC}}^{max}$ for different QoS sensitivity. Maximum QoS negotiation times differs from different service class and service type and is defined as $NT_{ST_{SC}}^{max}$. QoS monitor time interval for service class SC of service type ST is $MT_{Q_{ST_{SC}}}$ and QoS alert monitor time interval for service class SC of service type ST is $MT_{QA_{ST_{SC}}}$.

Table 3-7 QoS Requirement Interface

Field	Parameter	Unit/meaning
Service Type	ST	Video, VoIP, .
Service Class	SC	Platinum, gold, ...
QoS requirement	$QR_{ST_{SC}}$	Frames/second
Max. QoS alert times	$AT_{ST_{SC}}^{max}$	Times
Max. QoS negotiation times	$NT_{ST_{SC}}^{max}$	Times
QoS monitor time interval	$MT_{Q_{ST_{SC}}}$	Second
QoS alert monitor time interval	$MT_{QA_{ST_{SC}}}$	Second

3.6.10.5 Network Interface

Network interface defines the network resource requirement for external provider such as service provider and equipment provider. Service provider could subscribe with network provider to obtain enough network resource for providing better QoS such as minimum network throughput $TP_{ST_{SC}}^{min}$ and target bandwidth $TBW_{ST_{SC}}$. Target bandwidth is request to keep a number of bandwidth allocation for this session from starvation while congestion.

Table 3-8 Network Interface

Field	Parameter	Unit/meaning
Default DSCP	DSCP	EF, AF1, ...
Min. network throughput	$TP_{ST_{SC}}^{min}$	bps
Target bandwidth	$TBW_{ST_{SC}}$	bps

3.7 Mapping of Feedback-Based QoS control to UMTS

We extend or mapping our feedback-based QoS control framework to UMTS QoS management. The modified QoS management functions for controlling the UMTS bearer service are shown in Fig. 3-25. These control functions support the establishment and the modification of a UMTS bearer service by signaling/negotiation with the UMTS external services and by the establishment or modification of all UMTS internal services with the required characteristics.

3.7.1 Translation functions (Trans.)

Translation functions in the MT and the Gateway convert between external service signaling and internal service primitives including the translation of the service attributes. The translation function in the Gateway is FFS regarding packet oriented services. Our QoS Agent which converts the application's performance to QoS and feedback could be mapped in these functions or co-work with these functions 【6】.

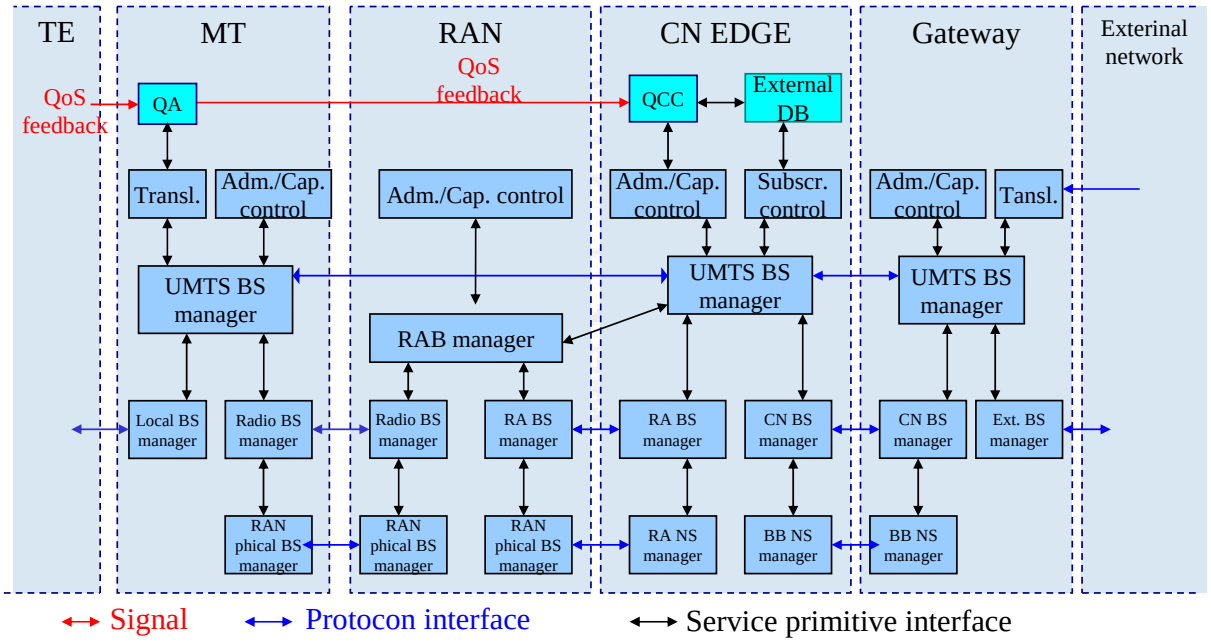

Fig. 3-25 Modified QoS management in control plane【6】

3.7.2 External DB

User profile management, SLA management and equipment management are integrated in an external DB functional block.

3.7.3 UMTS BS manager

The UMTS BS manager in the MT, CN EDGE and the Gateway signal between each other and via the translation function with external instances to establish or modify a UMTS bearer service. Each of the UMTS BS managers interrogates its associated admission/capability control whether the network entity supports the specific requested service and whether the required resources are available. Additionally, the CN EDGE UMTS BS manager verifies with the subscription control the administrative rights for using the service 【6】.

The UMTS BS manager of the MT translates the UMTS bearer service attributes into attributes for the local bearer service and requests this service from the local BS manager.

The UMTS BS manager of the CN EDGE translates the UMTS bearer service attributes into RAB service attributes and RAN Access bearer service attributes and it translates UMTS bearer service attributes into CN bearer service attributes. Also, the UMTS BS manager of the CN EDGE requests its RAN Access BS manager, its CN BS manager and the RAB manager in the RAN to provide the required services.

So UMTS BS manager is resource manager of UMTS and QoS control center could communicate with it for QoS negotiation. QoS control center could mapped into or co-work with administration/capacity control which is default function to communicate with UMTS BS manager in original UMTS QoS management in control plane.

3.7.4 RAB manager

The RAB manager verifies with its admission/capability control whether the RAN supports the specific requested service and whether the required resources are available. It translates the RAB service attributes into radio bearer service and RAN Access bearer service attributes and requests the radio BS manager and the RAN Access BS manager to provide bearer services with the required attributes 【6】.

RAB manager also is resource manager but manage lower level resource. Feedback-based QoS control may not need to contact with lower level manager directly but negotiates with high level resource manager and control low level network resource through high level resource manager to translate or communicate with low level resource manager.

3.7.5 Gateway UMTS BS manager

The Gateway UMTS BS manager translates the UMTS bearer service attributes into CN bearer service attributes and requests its CN BS manager to provide the service. Furthermore, it translates the UMTS bearer service attributes into the external bearer service attributes and requests this service from the external BS manager 【6】.

3.7.6 Other Lower Layers Manager

Radio, RAN Access and CN BS managers use services provided by lower layers and are described in 【6】.

3.7.7 Mapping to IP-Multimedia Subsystem (IMS)

UMTS especially specifies an IP-Multimedia Subsystem (IMS) for IP-multimedia control. We can take IMS as a subsystem of QoS management in UMTS. Via the IMS, UE negotiates its capabilities and express its QoS requirements during a Session Initial Protocol (SIP) session set-up or session modification procedure 【20】. The aim of using SIP is focus on IP connectivity and QoS is not the main point in SIP.

3GPP has decided to use a layered approach to architectural design which means transport and bearer services are separated from the IMS signaling network and session management services. Future services are run on top of the IMS signaling network. The idea of feedback-based QoS control framework refers to IMS and separates the QoS signaling and QoS control from network level. We could easily maps feedback-based QoS control framework to IMS or extend or co-work with the functions in IMS such as Fig. 3-26 shows.

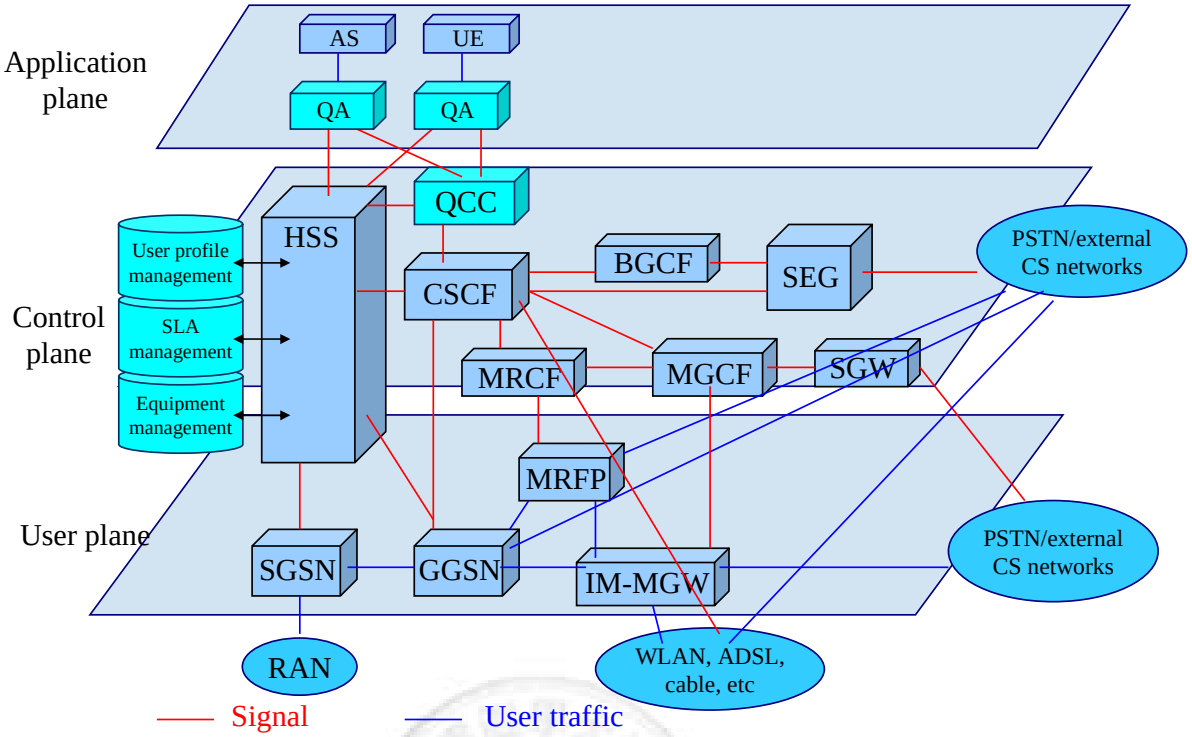

Fig. 3-26 Modified IMS architecture 【20】

Application Server (AS) and UE could embedded with or install with QoS Agent (QA) to monitor application's performance. QoS is feedback to QoS Control Center (QCC). QCC could co-work with Call Session Control Function (CSCF) or be implemented in CSCF. CSCF manage sessions which generates flows and we could control QoS per-flow via QCC and CSCF. QCC communicate with CSCF for QoS control based on QoS feedback.

Home Subscriber Server (HSS) is the main data storage for all subscriber and service-related data such as user profile, SLA and equipment profile. The user profile management, SLA management and equipment could be placed in HSS or co-work with HSS to manage user profile, SLA and equipment profile. QCC could obtain and maintain service-related data through HSS.

Other entities in IMS which we not described in thesis because beyond our research scope are detail described in 【20】.