Simulation model of a user-manageable quality of service control method

Karol Molnar

Dept. of Telecommunications, FEEC
Brno University of Technology, Purkynova 118, Brno, Czech Republic
molnar@feec.vutbr.cz

Abstract. Diffserv, the most often used Quality of Service support technology, offers fast and efficient data flow processing. On the other hand these advantages are achieved at the cost of static, network-driven data-flow metering, classification, marking and queue management, which does not support any interaction with the end-user services. The following work offers a solution how to allow the end-user or end-system to cooperate with the edge router of a DiffServ domain and further increase the efficiency of the available QoS-support system.

Keywords: QoS, DiffServ, MIB, SNMP

1 Introduction

Modern packet-switched network technologies are designed to support different network services and, under a moderate network load, these services can offer an appropriate level of service quality. In the network nodes the incoming data flow is usually classified into a relatively small number of service classes and each class is assigned a specific queue. The queues are then served by a deterministic packet scheduler, very often implementing the Priority Queuing (PQ) or the Weighted Fair Queuing (WFQ) mechanism. This solution offers stateless and so far very fast traffic differentiation and differentiated packet processing.

The above principles are also implemented in the technology of Differentiated Services (DiffServ), which is nowadays the most extended solution for Quality of Service (QoS) support. Because of its relative simplicity and stateless packet processing, DiffServ offers high scalability and can efficiently operate in large data networks too. This is the reason why DiffServ is practically the only world-wide extended QoS support technology in use.

A serious limitation of the DiffServ technology is its network-oriented data-flow processing. All packet-processing rules, including service classification, traffic metering, packet marking, queuing and packet scheduling, are configured in network elements, usually in the edge and the core routers, and operate totally independently of the user stations.

The missing interaction between DiffServ network components and the user terminal represents a severe gap, which greatly conduces to the slow growth of QoS–support

technology implementations. By allowing an application to define its requirements on network resources and specify the desired service class, this situation could be changed. However, this improvement requires the accomplishment of two main requirements. First, the networking application must be able to define its requirements on the network resources. Since the QoS support is required mainly in multimedia applications, with both conversional and streaming character of traffic, the prediction of the required resources is directly related to the type and parameters of the codecs used and usually can be specified quite precisely. In the second step, the requirements specified must be communicated to the network, more precisely to the QoS support technology used in this network. The objective of our work is to offer a solution for this communication.
Since the DiffServ specification does not reckon with direct control mechanisms between the edge router and the end end-station, there is no dedicated communication protocol designed for this purpose. In addition, the edge router is not designed to offer extensive control functions to the end-stations. In order not to burden the edge router, such an extensive communication should be avoided.
The solution presented in this document tries to affect the classical DiffServ specification as little as possible. Instead of trying to control the edge router we focus on how to retrieve DiffServ-related configuration data from this router and transform this information into a form easily understandable to the end-user or an application. Based on this information the user or the application can automatically select the service class which best suits its requirements.
We divided the experimental implementation of the mechanism presented into several phases. First, the suggested method is evaluated in the OPNET Modeler simulation environment. Since the OPNET Modeler uses a C-based programming language to describe the models on the process level, the majority of the algorithms implemented in the simulation model can also be used for practical evaluation in a laboratory environment. This document presents the results of the theoretical preparation work.

2 The communication model

The key component of the user-manageable QoS control method mentioned above is an information exchange between the end-station and the edge router. As a result of this process the end-station can obtain a limited set of DiffServ configuration parameters from the edge routers. It would be possible to design a proprietary protocol for this purpose, but using the standardized Simple Network Management Protocol (SNMP) [4] offers the same results with much less effort. In addition, the SNMP manager, through the built-in SNMP agent, has direct access to the majority of (if not all) configuration parameters of the edge router, and no further tool is required to obtain these data. Using the SNMP can greatly simplify the situation and allow us to concentrate on the basic functionalities of the suggested extension.
For security reasons, normal users are usually not allowed to access directly the SNMP agent. On the other hand, our solution requires just read-only access and only to a clearly specified part of the Management Information Base (MIB) [5]. The amount of data, that should be accessible, can thus be largely reduced.

The communication process used in the suggested user-manageable quality of service control method is shown in Fig.1.

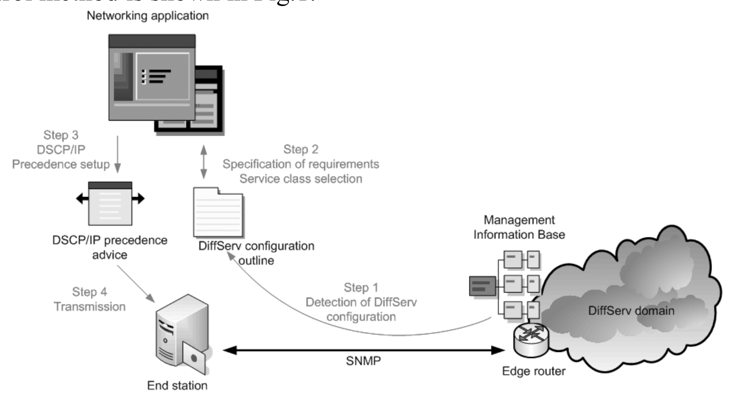

Fig. 1. Concept of the user-manageable quality-of-service control method

First, the end-station must detect the current DiffServ configuration by retrieving the related data from the edge router's MIB. The method suggested uses the SNMP to collect this information from the edge routers. The configuration is usually accessible from a special MIB controlled by the SNMP agent implemented in the router, so the end station with sufficient access rights could retrieve this information.
In the next step, the acquired information must be processed and transformed into a form that the user-application can simply evaluate. Based on the information about the available service classes and their parameters, the application automatically, or in cooperation with the end-user, can select the most suitable service class. The desired service class can be indicated by setting the corresponding DiffServ Code Point (DSCP) value in the header of the transmitted IP packets. These packets with the desired DSCP value are then sent to the edge router. Packet processing in the edge router will be realized in a standard way, meaning that the traffic is first classified and then metered. If there are enough network resources and the incoming traffic flow does not violate the Service Level Agreement (SLA), the edge router can keep the demanded service class for the data flow. If there is any conflict, the packet marking will be based on the results of the measurement process and the desired DSCP will be overwritten with a value determined by the edge router. Using this solution the final decision will be made by the edge router so that the DiffServ domain will not be compromised by occasionally or intentionally misclassified traffic.

2.1 Simplified SNMP manager

We have defined that the SNMP protocol will be used between the end-station and the edge router to obtain the DiffServ configuration. Of course, our solution does not require a complex SNMP manager full of features. Rather a small and fast manager application should be used which is able to connect to one specific device, the edge router, and can acquire a part of the implemented MIBs and can do this task repeatedly with a relatively long repetition interval. According to our requirements it is enough to support only three SNMP operations, in particular GetRequest, GetNextRequest and GetResponse, in this manager application. For the purpose of simulation it is not really important, but for practical realization we want to assign this functionality to a service or daemon running directly under the operating system.

2.2 DiffServ-related MIBs

The most crucial part of the designed system is represented by the Management Information Base containing the required DiffServ configuration parameters. The complications come from the fact that there is not a single universal DiffServ-MIB used in all network elements. The Internet Engineering Task Force published a Request For Comment 3289 [1] containing a concept for DiffServ-MIB, but practically it is not in use. The manufacturers usually define their own proprietary solutions. The next chapters shortly describe the DiffServ MIB designed by IETF and the Class- based QoS MIB used by Cisco Systems Inc.

DiffServ Informal Management Model

In the year 2002, IETF published An Informal Management Model for DiffServ Routers in RFC 3290 [2]. This document introduces a management model which could be used for modelling management and configuration tasks in DiffServ routers. The management model includes all elementary components of the DiffServ technology, called DiffServ data path elements in the document. The model of each data path element contains the corresponding configuration parameters and a way of linking them with other elements. The cascade of these elements builds up a Traffic Conditioning Block (TCB). These modelled elementary components are:

- Classifiers (e.g. Behaviour Aggregate and Multi-Field Classifier),
- Meters (e.g. Average Rate, Exponential Weighted Moving Average, and several Token Bucket Meters),
- Action elements (e.g. DSCP Marker, Absolute Dropper, Multiplexor or Counter),
- Queuing elements (e.g. FIFO Queue, scheduler, algorithmic Dropper, etc.).

The Traffic Conditioning Block thus represents a set of data path elements providing specific traffic treatment. There is no fixed position for the data path elements in the TCB. They can be linked together arbitrarily according to the desired traffic policy. An example of a TCB [2] is shown in Fig.2.
In spite of the complexity of the TCB shown in the Figure its functionality is quite evident from the blocks used. It represents a exemplary way of processing data flow, starting with packet classification and followed by the measurement of different traffic classes, marking them accordingly, adding selective discard to prevent router form congestions, storing the processed packets in a queue, and scheduling packet transmission from the queues.

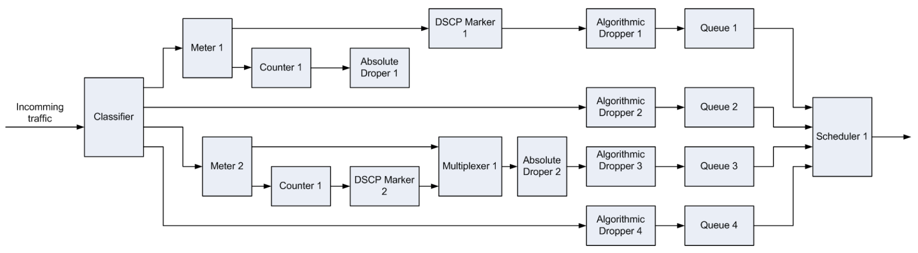

Fig. 2. Example of a Traffic Conditioning Block

IETF DiffServ MIB

Based on the DiffServ Informal Management Model, IETF defined the DiffServ MIB designed for configuration and management purposes. The DiffServ MIB is designed to allow both remote monitoring and configuration of DiffServ routers. The Object Identifier of the MIB is 1.3.6.1.2.1.97 and is located under the iso.org.dod.internet.mgmt.mib-2.diffServMIB tree. There are three basic branches defined in the MIB: diffServMIBObjects, diffServMIBConformance and diffServMIBAdmin. The diffServMIBObjects branch contains the management structure for each data path element. The diffServMIBConformance branch defines the MIB statements for full and for read-only compliance implementations and defines the MIB groups which should be implemented. The third branch, diffServMIBAdmin, defines a management model for concrete implementations of token-bucket meters and schedulers.

The management models of data path elements are stored in tables and the related table entries are linked together using pointers called RowPointer. The data path elements are divided into the following tables:

Data path table (diffServDataPathTable) defines the starting point of the data path identifying the interface, its direction and assigning a pointer pointing to the first functional data path element.

Classifier table (diffServClassifier) stores information about classifier elements, which are used to identify a specific part of the network traffic. In RFC 3289 [2] only the Multifield Classifier is implemented but the pointer-based structure is flexible enough to define additional types. The traffic, according to the filtering criteria, is detached and then sent to the next data path element.

Meter table (diffServMeter) contains the DiffServ traffic meters used in the router. Meters have two outputs, one for the traffic conforming to the metering parameters and one for the traffic exceeding these parameters. The next data path element for both directions is identified by a separate RowPointer.

Token-bucket parameter table (diffServTBParam) contains the list of token-bucket meters which can be implemented in the metering elements.

Action table (diffServAction) contains the data path elements which provide different actions applied to the traffic. These actions are DSCP marking or packet counting. Packet counting is used mainly for statistics and measurement. After providing the required action, packets are forwarded the next data path element identified by a pointer.

Algorithmic dropper table (diffServAlgDrop) contains controllable droppers used to prevent queues from being overloaded. RFC 3289 [2] defines a management model for several dropping algorithms, like tail drop, head drop or parametric random drop algorithms.

Queue table (diffServQueue) stores information about the queues implemented in the router. Each queue is modelled by a FIFO queue, but using a scheduler element the FIFO queues can be organized into a complex queue system.

Scheduler Table (diffServScheduler) contains the list of scheduler elements which are responsible for packet scheduling and are managing the related queues according to the configured relationships between them.

The IETF´s DiffServ MIB offers very flexible monitoring and configuration of DiffServ elements, but on the other hand it is quite complex. The biggest disadvantage is that it is not widely used by the manufacturers of active network elements. As an alternative encountered in practice we can mention the Cisco Class based QoS MIB.

Cisco Class-based QoS MIB

The Cisco Class-based QoS MIB is a proprietary Management Information Base for Cisco devices supporting the Modular QoS Command-line Interface. In contrast to the DiffServ MIB, the main purpose of the Class-based QoS MIB is to provide read access to QoS configuration in the DiffServ router and allow the network manager to collect statistical information about the traffic processed.

The Class-based QoS MIB uses twenty tables to describe configurations and statistics for the implemented traffic policies, class mappings, classifiers and queues, for packet marking and for the Random Early Detection algorithm. One of the tables, called QoS objects, is used to collect all the implemented components of the DiffServ model. The QoS objects table consists of ClassMaps, Match Statements, PolicyMaps and Feature Actions.

- The **Match** statement specifies match criteria which are used to identify packets for classification purposes.
- The **ClassMap** object represents a user-defined traffic class that contains one or more match statements used to classify packets into several categories.
- The **Feature Action** represents the way the selected part of traffic is processed. Features include policing, traffic-shaping, queuing, random dropping using the RED method, and packet marking.
- The **PolicyMap** table contains the associations between the QoS action and the traffic class defined in the ClassMap table.

The process of reading information form the MIB starts by learning the cbQosServicePolicyTable and cbQosObjectsTable MIB tables. The corresponding indexes are cbQosPolicyIndex and cbQosObjectsIndex. cbQosPolicyIndex is designed to identify the service policies attached to logical interfaces and the cbQosObjectsIndex is designed to identify each QoS feature on a specified device. The DiffServ related configuration parameters are stored in the system corresponding to the structure of command-line commands.

2.3 The simulation model

The simulation model for the user-manageable quality of service control method proposed in this document is developed in the OPNET Modeler environment. The OPNET Modeler is an up-to-date simulation environment capable of simulating the behaviour of network processes (communication protocols), network components (servers, workstations, switches, routers, etc.), applications (http, ftp, email, VoIP, database, etc.) and their extended combinations (subnetworks, fixed and wireless networks, etc.). It also supports Differentiated Services with the configuration process quite similar to the configuration of real systems.

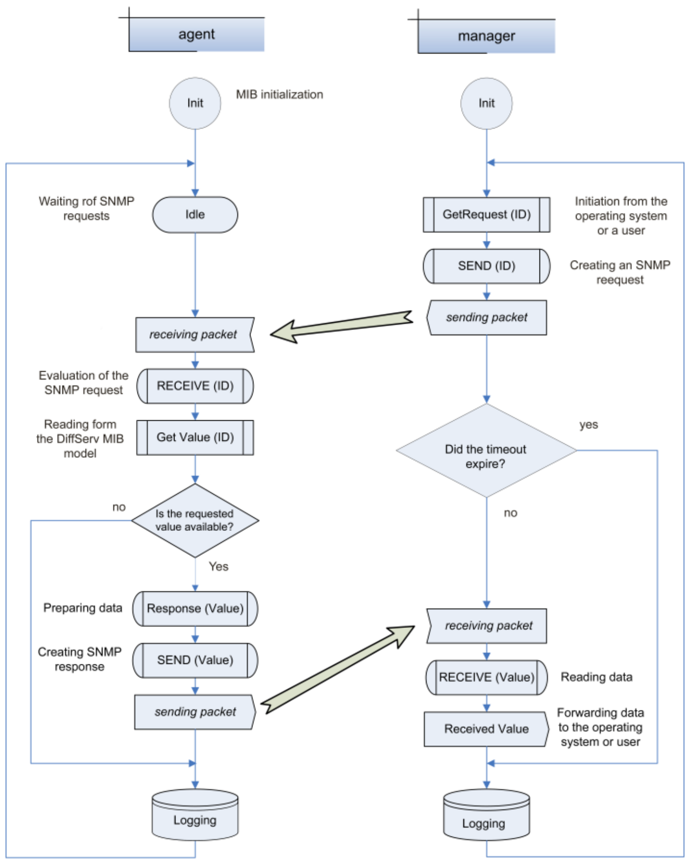

Fig. 3. The state-machine of the manager and agent

Since, the structure of the Cisco Class-based QoS MIB is simpler and it is more suitable for practical implementation, we decided to use IETF´s DiffServ MIB in our model. Since we need to simulate only a limited part of the DiffServ model, we could simplify enough the DiffServ MIB to implement it into our simulation model. Our system requires information only about the available service classes metering parameters and the corresponding queuing system. We do not need to deal with packet dropping, processing of packets not conforming to the metering criteria and we do not require statistics in our model. Using these simplifications a relatively compact model of DiffServ MIB can be defined.

On the other hand, to implement the simulation model we had to redefine the standard application part available in the process model of a workstation and server. Since the SNMP is not supported in the current version of the OPNET Modeler (version 12.0 Patch Level 5) we had to define a new SNMP application and link it with the UDP transport protocol. For this purpose we used the Interface Control Information (ICI) [6] functions, which are able to build virtual connections between different network entities. The state machine of the implemented agent and simplified manager are shown in Fig. 3.

At the time of writing this document the functional evaluation is in progress. The data exchange between the models of workstation and server is evaluated together with the algorithms working with the simplified MIB.

3 Conclusion

The technology of Differentiated Services can significantly improve the efficiency of network transmissions if it is correctly configured. On the other hand, DiffServ is not able to directly interact with the end-user applications and that is why it is not able to support end-to-end QoS. In this document a user-manageable QoS control method is introduced which partially overcomes this limitation. For the communication between the edge router and the end-station a standardized SNMP protocol has been chosen. As a result of this communication the end-station is able to obtain the current DiffServ configuration and the application can suggest for the data to be sent the service class which fits its requirements the most.

In the current state of realization a theoretical preparation has been finished and the building of a simulation model on OPNET Modeler is in progress. By now the standard process models of a workstation and a server have been extended with a simplified version of an SNMP agent and manager, respectively. The communication between these components is solved using the Interface Control Information API. Functions working with the MIB are also implemented. Some complications were caused by the fact that there is no general DiffServ-related MIB defined and each manufacturer uses their own proprietary solution. After several evaluations we chose for our simulation model the DiffServ MIB defined by the Internet Engineering Task Force.

The future work will be concentrated on finishing the simplified DiffServ MIB implementation and optimizing the communication between the end-station and the edge router. After this task is successfully finished, we will start to work on the evaluation in laboratory environment.

Acknowledgement: This paper has been supported by the Grant Agency of the Czech Republic (grants No. 102/06/1569 and No. 102/05/P585), Ministry of Industry and Trade of the Czech Republic (FT-TA3/011) and the Ministry of Education of the Czech Republic (grants No. 1K04116 and No. MSM0021630513).

References

1. Bernet, Y., Blake, S., Grossman, D., Smith, A.: An Informal Management Model for Diffserv Routers, RFC 3290, Internet Engineering Task Force (2002)
2. Baker, F., Chan, K., Smith, A.: Management Information Base for the Differentiated Services Architecture, RFC 3289, Internet Engineering Task Force (2002)
3. Fung, A.: Cisco Class-Based QoS MIB, Cisco Systems Inc., http://tools.cisco.com/Support/SNMP/do/BrowseMIB.do?local=en&mibName=CISCO-CLASS-BASED-QOS-MIB-CAPABILITY (2001)
4. Case, J., McCloghrie, K., Rose, M., Waldbusser, S.: Structure of Management Information for version 2 of the Simple Network Management Protocol (SNMPv2) RFC 1442, Internet Engineering Task Force (1993)
5. Case, J., McCloghrie, K., Rose, M., Waldbusser, S.: Management Information Base for version 2 of the Simple Network Management Protocol (SNMPv2) RFC 1450, , Internet Engineering Task Force (1993)
6. Opnet Technologies, Inc.: OPNET Modeler Product Documentation Release 12.0, OPNET Modeler, 2006