IETF Internet Draft NSIS Working Group
Internet Draft
<draft-ietf-nsis-qspec-03.txt>
Expiration Date: August 2005

Jerry Ash
AT&T
Attila Bader
Ericsson
Cornelia Kappler
Siemens AG

February 2005

QoS-NSLP QSPEC Template

Status of this Memo

Copyright Notice

Abstract

The QoS NSLP protocol is used to signal QoS reservations and is independent of a specific QoS model (QOSM) such as IntServ or DiffServ. Rather, all information specific to a QOSM is encapsulated in a separate object, the QSPEC. This draft defines a template for the QSPEC, which contains both the QoS description and QSPEC control information. The QSPEC format is defined, as are a number of QSPEC parameters. The QSPEC parameters provide a common language to be re-used in several QOSMs, which are derived initially from the IntServ and DiffServ QOSMs. To a certain extent QSPEC parameters

ensure interoperability of QOSMs. Optional QSPEC parameters aim to ensure the extensibility of QoS NSLP to other QOSMs in the future. The node initiating the NSIS signaling adds an Initiator QSPEC that must not be removed, thereby ensuring the intention of the NSIS initiator is preserved along the signaling path.

Table of Contents

1. Conventions Used in This Document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2. Introduction

The QoS NSLP establishes and maintains state at nodes along the path of a data flow for the purpose of providing forwarding resources (QoS) for that flow [QoS-SIG]. The design of QoS NSLP is conceptually similar to RSVP [RFC2205], and meets the requirements of [RFC3726].

A QoS-enabled domain supports a particular QoS model (QOSM), which is a methodology to achieve QoS for a traffic flow that incorporates QoS provisioning methods and a QoS architecture. A QOSM defines the behavior of the resource management function (RMF), including inputs and outputs, and how QSPEC information is interpreted on traffic description, resources required, resources available, and control information required by the RMF. A QOSM also specifies a set of mandatory and optional QSPEC parameters that describe the QoS and how resources will be managed by the RMF. QoS NSLP can support signaling for different QOSMs, such as for IntServ, DiffServ admission control, and those specified in [Y.1541-QOSM, INTSERV-QOSM, RMD-QOSM]. For more information on QOSMs see Appendix B.

One of the major differences between RSVP and QoS-NSLP is that QoS-NSLP supports signaling for different QOSMs along the data path, all with one signaling message. For example, the data path may start in a domain supporting DiffServ and end in a domain supporting Y.1541. However, because some typical QoS parameters are standardized and can be reused in different QOSMs, some degree of interoperability between QOSMs exists.

The QSPEC object travels in QoS-NSLP messages and is opaque to the QoS NSLP. The content of the QSPEC object is QOSM specific. Since QoS-NSLP signaling operation can be different for different QOSMs, the QSPEC contains two kinds of information, QSPEC control information and QoS description.

QSPEC control information contains parameters that governs the RMF. An example of QSPEC control information is how the excess traffic is treated in the RMF queuing functions.

The QoS description is composed of objects loosely corresponding to the TSpec, RSpec and AdSpec objects specified in RSVP. This is, the QSPEC may contain a description of QoS desired and QoS reserved. It can also collect information about available resources. Going beyond RSVP functionality, the QoS description also allows indicating a range of acceptable QoS by defining an object denoting minimum QoS. Usage of these objects is not bound to particular message types thus allowing for flexibility. An object collecting information about available resources MAY travel in any QoS NSLP message, for example a QUERY message or a RESERVE message.

3. Terminology

Mandatory QSPEC parameter: QSPEC parameter that a QNI SHOULD populate if applicable to the underlying QOSM and a QNE MUST interpret, if populated.

Minimum QoS: Minimum QoS is a functionality that MAY be supported by any QNE. Together with a description of desired QoS, it allows the QNI to specify a QoS range, i.e. an upper and lower bound. If the desired QoS is not available, QNEs are going to decrease the reservation until the minimum QoS is hit.

Optional QSPEC parameter: QSPEC parameter that a QNI SHOULD populate

if applicable to the underlying QOSM, and a QNE SHOULD interpret if populated and applicable to the QOSM(s) supported by the QNE. (A QNE MAY ignore if it does not support a QOSM needing the optional QSPEC parameter).

QNE: QoS NSIS Entity, a node supporting QoS NSLP.

QNI: QoS NSIS Initiator, a node initiating QoS NSLP signaling.

QNR: QoS NSIS Receiver, a node terminating QoS NSLP signaling.

QoS Description: Describes the actual QoS in objects QoS Desired, QoS Available, QoS Reserved, and Minimum QoS. These objects are input or output parameters of the RMF. In a valid QSPEC, at least one object of the type QoS Desired, QoS Available or QoS Reserved MUST be included.

QoS Available: Object containing parameters describing the available resources. They are used to collect information along a reservation path.

QoS Desired: Object containing parameters describing the desired QoS for which the sender requests reservation.

QoS Model (QOSM): A methodology to achieve QoS for a traffic flow, e.g., IntServ Controlled Load. Specifies a set of mandatory and optional QSPEC parameters that describe the QoS and how resources will be managed by the RMF.

QoS Reserved: Object containing parameters describing the reserved resources and related QoS parameters, for example, bandwidth.

QSPEC Control Information: Control information that is specific to a QSPEC, and contains parameters that govern the RMF.

QSPEC: QSPEC is the object of QoS-NSLP containing all QOSM specific information.

QSPEC parameter: Any parameter appearing in a QSPEC; includes both QoS description and QSPEC control information parameters, for example, bandwidth, token bucket, and excess treatment parameters.

QSPEC object: Main building blocks of QoS Description containing a

parameter set that is input or output of an RMF operation.

Resource Management Function (RMF): Functions that are related to resource management, specific to a QOSM. It processes the QoS description parameters and QSPEC control parameters.

Read-only Parameter: Parameter that is set by initiating or responding QNE and is not changed during the processing of the QSPEC along the path.

Read-write Parameter: Parameter that can be changed during the processing of the QSPEC by any QNE along the path.

4. QSPEC Parameters, Processing, & Extensibility

4.1 QSPEC Parameters

The definition of a QOSM includes the specification of how the requested QoS resources will be described and how they will be managed by the RMF. For this purpose, the QOSM specifies a set of QSPEC parameters that describe the QoS and QoS resource control in the RMF. A given QOSM defines which of the mandatory and optional QSPEC parameters it uses, and it MAY define additional optional QSPEC parameters. Mandatory and optional QSPEC parameters provide a common language for QOSM developers to build their QSPECs and are likely to be re-used in several QOSMs. Mandatory and optional QSPEC parameters are defined in this document, and additional optional QSPEC parameters
can be defined in separate documents. Specification of additional optional QSPEC parameters requires standards action, as defined in Section 4.5.

4.2 QSPEC Processing

The QSPEC is opaque to the QoS-NSLP processing. The QSPEC control information and the QoS description are interpreted and MAY be modified by the RMF in a QNE (see description in [QoS-SIG]).

A QoS-enabled domain supports a particular QOSM, e.g. DiffServ. If this domain supports QoS-NSLP signaling, its QNEs MUST support the DiffServ QOSM. The QNEs MAY also support additional QOSMs.

A QoS NSLP message can contains a stack of 1+n QSPECs. The first on the stack is the Initiator QSPEC. This is a QSPEC provided by the QNI, which travels end-to-end, and therefore the stack always has at least depth 1. QSPEC parameters MUST NOT be deleted from or added to the Initiator QSPEC. In addition, the stack MAY contain n Local QSPECs stacked on top of the Initiator QSPEC, where n is the level of nested QOSM regions. In most cases, we expect the QSPEC stack to be no deeper than 2. A QNE only considers the topmost QSPEC.

At the ingress edge of a local QoS domain, a Local QSPEC MAY be pushed on the stack in order to describe the requested resources in a domain-specific manner. Also, the Local QSPECs are popped from the stack at the egress edge of the local QoS domain.

This draft provides a template for the QSPEC, which is needed in order to help defining individual QOSMs and in order to promote interoperability between QOSMs. Figure 1 illustrates how the QSPEC is composed of QSPEC control information and QoS description. QoS description in turn is composed of up to four objects (not all of them need to be present), namely QoS desired, QoS available, QoS reserved and Minimum QoS. Each of these objects, as well as QSPEC control information, consists of a number of mandatory and optional QSPEC parameters.

```
   +------------------------+----------------------------------------+
   |  QSPEC control info    |             QoS description            |
   +------------------------+----------------------------------------+

                             \________________ ______________________/
                                              V
                            +----------+----------+----------+-------+
                            |QoS Desir.|QoS Avail. |QoS Rsrv. |Min QoS|
                            +----------+----------+----------+-------+

   \__________ ____________/\_____ ____/\___ ______/\___ _____/\__ ___/
              V                  V         V           V         V

    +-------------+...      +-------------+...
    |QSPEC para. 1|         |QSPEC para. n|
    +-------------+...      +-------------+...
```

Figure 1: Structure of the QSPEC

The Initiator QSPEC additionally MAY contain optional QSPEC parameters in each object and in the QSPEC control information, as illustrated in Figure 2.

```
+----------------+------------------------------+------------------------
-+
   |QSPEC Object ID|  Mandatory QSPEC Parameters  |Optional QSPEC
Parameters|
```

```
+---------------+------------------------------+------------------------
-+
```

Figure 2: Structure of Initiator & Local QSPEC Objects & Control Information

4.3 Example of NSLP/QSPEC Operation

This Section illustrates the operation and use of the QSPEC within the NSLP. The example configuration in shown in Figure 3.


```
+----------+      /-------\       /--------\       /--------\
| Laptop   |      |  Home  |      |  Cable  |      | Diffserv |
| Computer |-----| Network |-----| Network  |-----| Network  |----+
+----------+      | No QOSM |     |DQOS QOSM |      | RMD QOSM |    |
                  \-------/       \--------/       \--------/     |
                                                                  |
                  +-----------------------------------------------+
                  |
                  |    /--------\       +----------+
                  |    |  "X"G   |      | Handheld |
                  +---| Wireless |-----|  Device  |
                       | XG QOSM |      +----------+
                       \--------/
```

Figure 3: Example Configuration to Illustrate NSLP/QSPEC Operation

In this configuration, a laptop computer and a handheld wireless device are the end points for some application that has QoS requirements. Assume that the two end points are stationary during the application session. For this session, the laptop computer is connected to a home network that has no QoS support. The home network is connected to a CableLabs-type cable access network with dynamic QoS (DQOS) support, such as specified in the 'CMS to CMS Signaling Specification' [CMSS] for cable access networks. That network is connected to a Diffserv core network that uses the RMD QOSM. On the other side of the Diffserv core is a wireless access network built on generation "X" technology with QoS support as defined by generation "X". And finally the handheld endpoint is connected to the wireless access network.

We assume that the Laptop is the QNI and Handheld Device is the QNR.

The QNI will populate an Initiator QSPEC to achieve the QoS desired on the path. There are two cases to consider:

Case 1) The QNI knows, either by discovery or configuration, the QOSMs supported in the downstream domains, and populates the appropriate mandatory and optional QSPEC parameters to achieve the QNI's desired QoS.

Case 2) The QNI does not know which QOSMs are supported in the downstream domains, and populates appropriate mandatory and optional QSPEC parameters so that downstream QNEs can interpret the parameters to achieve the QNI's desired QoS.

In case 1, the Laptop-QNI discovers which QOSMs are supported on the data path by sending a QUERY message along the data path. The RESPONSE message indicates the preferred QOSM supported in the domains along the path. The Laptop-QNI then knows that the RMD-QOSM and XG-QOSM are in the path and populates the appropriate mandatory and optional QSPEC parameters needed by the RMD-QSPEC and XG-QSPEC in the Initiator QSPEC sent in the RESERVE message.

In case 2, the QNI populates mandatory and optional QSPEC parameters to ensure correct treatment of its traffic in domains down the path. Since the QNI does not know the QOSM used in downstream domains, it

includes values for all mandatory and optional QSPEC parameters it cares about. For example, if the QNI wants to achieve IntServ-like QoS guarantees, it would include parameters for IntServ Guaranteed Service. Since these parameters give a very precise description of what is desired, QNEs can do their best to satisfy the request. If the QNI thinks DiffServ priority treatment would be sufficient, it just includes the appropriate DiffServ parameters, for example, it identifies the EF PHB in the <QoS Class> parameters.

In both cases, each QNE on the path reads and interprets the parameters in the Initiator QSPEC that it needs to implement the QOSM within its domain. For example, at the RMD and XG network ingress, a Local QSPEC corresponding to the RMD-QOSM and XG-QOSM are generated, respectively, according to the procedures described in Section 4.5 of [QoS-SIG]. That is, the RMD-QNI and XG-QNI must map the mandatory and optional QSPEC parameters contained in the Initiator QSPEC into the appropriate Local QSPEC objects to be pushed on top of the Initiator QSPEC within the RMD and XG domains,

respectively. For example, in the RMD domain the <Bandwidth> parameter is read if available from the Initiator QSPEC, or alternatively it interprets the needed bandwidth parameter based on <Token Bucket> parameters in the Initiator QSPEC.

This top Local QSPEC becomes the current QSPEC used within the RMD and XG domain, that is, the translated Initiator QSPEC becomes the first (Local) QSPEC, and the Initiator QSPEC is second. This saves the QNEs within the RMD and XG domains the trouble of re-translating the Initiator QSPEC. At the egress edge of the RMD and XG domains, the translated QSPEC is popped, and the Initiator QSPEC returns to the number one position. Eventually the RESERVE request arrives at the QNR.

4.4 Treatment of QSPEC Parameters

Mandatory and optional QSPEC parameters are defined in this document and are applicable to a number of QOSMs. Mandatory QSPEC parameters are treated as follows:

- o A QNI SHOULD populate mandatory QSPEC parameters if applicable to the underlying QOSM.
- o QNEs MUST interpret mandatory QSPEC parameters, if populated.

Optional QSPEC parameters are treated as follows:

- o A QNI SHOULD populate optional QSPEC parameters if applicable to the underlying QOSM.
- o QNEs SHOULD interpret optional QSPEC parameters, if populated and applicable to the QOSM(s) supported by the QNE. (A QNE MAY ignore if it does not support a QOSM needing the optional QSPEC parameter).

4.5 QSPEC Extensibility

Additional optional QSPEC parameters MAY need to be defined in the future. Additional optional QSPEC parameters are defined in separate Informational documents specific to a given QOSM.

5. QSPEC Format Overview

QSPEC = <QSPEC Control Information> <QoS Description>

As described above, the QSPEC object contains the actual resource description (QoS description) as well as QSPEC control information. Both QoS description and QSPEC control information MAY contain read-write and read-only objects.

Read-write objects can be changed by any QNE, whereas read-only objects are fixed by the QNI and/or QNR. An example of a read-write object is <QoS Available>, which is updated by intermediate QNEs. An example of a read-only object is <QoS Desired> as sent by the QNI.

Note that all QSPEC parameters defined in the following Sections are mandatory QSPEC parameters unless specifically designated as optional QSPEC parameters.

5.1. QSPEC Control Information

QSPEC control information is used for signaling QOSM RMF functions not defined in QoS-NSLP. It enables building new RMF functions required by a QOSM within a QoS-NSLP signaling framework, see for example [RMD-QOSM] and [Y.1541-QOSM].

```
<QSPEC Control Information> = <NON NSLP Hop> <NSLP Hops>
                              <Max NSLP Hops> <Excess Treatment>
```

This is a read-write object.

<NON NSLP Hop> is a flag bit telling the QNR (or QNI in a RESPONSE message) whether or not a completely reservation-capable path exists between the QNI and QNR. If the QNR finds this bit set, at least one QNE along the data transmission path between the QNI and QNR can not support QoS-NSLP/QSPEC at all. If this bit is set, the values of all other parameters in the <QoS Available> are unreliable.

<NSLP Hops> parameter indicates the number of QoS NSLP peers along the path the QNE MUST support and characterize the service in conformance with the relevant QoS-NSLP/QSPEC specification, and if it does not it MUST correctly set the <NON NSLP Hop> flag parameter. The composition rule for this parameter is to increment the counter by one at each QoS-NSLP/QSPEC-aware hop. This quantity, when composed from the QNI to QNR, informs the QNR (or QNI in a RESPONSE message) of the number of QoS-NSLP/QSPEC-aware QNEs traversed along the path. In a local QSPEC, <NSLP Hops> and <NON NSLP hop> refer to the NSLP nodes of the local QOSM.

<Max NSLP Hops>parameter limits the maximum number of QoS-NSLP hops. If <Max NSLP Hops> is reached, the NSLP message SHOULD be terminated.

<Excess Treatment> parameter describes how the QNE will process excess traffic, that is, out-of-profile traffic. Excess traffic MAY be dropped, shaped and/or remarked. The excess treatment parameter is initially set by the QNI and adjusted as needed by QNEs.

5.2 QoS Description

The QoS Description is broken down into the following objects:

<QoS Description> = <QoS Desired> <QoS Available> <QoS Reserved>
<Minimum QoS>

Of these objects, QoS desired, QoS available and QoS reserved MUST be supported by QNEs. Minimum QoS MAY be supported. QoS Desired and Minimum QoS are read-only, whereas QoS Available and QoS Reserved are read-write. If it needs to be ensured that QoS Desired and Minimum QoS are indeed not changed along the path, it is possible to apply selective protection of these objects only. The verification is based on cryptographic procedures.

On the QSPEC template level, the only restriction on object usage is that <Minimum QoS> SHOULD always travel together with <QoS Available> and/or <QoS Desired>. Otherwise there is no restriction on how many of these objects a QSPEC may carry, nor what type of object is carried in what type of QoS NSLP message. For example, in a receiver-initiated reservation scenario, the QNI MAY send a QUERY carrying a <QoS Available> object to probe the available resources on the path. The same QUERY MAY carry a <QoS Desired> object. The responding QNE can re-use the latter objects in the RESERVE message. A QNI could send a RESERVE with <QoS Desired> containing a particular <Bandwidth> parameter, and at the same time include a <QoS Available> object for querying availability of this same parameter. If <QoS Desired> cannot be reserved, the QNR can use the information collected in <QoS Available> along the path to signal back to the QNI a more promising value of <QoS Desired>. The details of such message exchanges are defined in [QoS-Sig]. The QoS NSLP and particularly the QOSMs prescribe how the objects in QSPECs are interpreted and used, and therefore MAY restrict this freedom.

5.2.1 <QoS Desired>

<QoS Desired> = <Traffic Description> <QoS Class> <Priority>

These parameters describe the resources the QNI desires to reserve and hence this is a read-only object. <QoS Desired> includes a description of the traffic the QNI is going to inject into the network.

<Traffic Description> = <Bandwidth> <Token Bucket>

<Bandwidth> = link bandwidth needed by flow [RFC 2212, RFC 2215]

<Token Bucket> = <r> <b> <p> <m> <M> [RFC 2210]

<QoS Class> = <PHB Class> <Y.1541 QoS Class> <DSTE Class Type>

An application MAY like to reserve resources for packets with a particular QoS class, e.g. a DiffServ per-hop behavior (PHB) [RFC2475], or DiffServ-enabled MPLS traffic engineering (DSTE) class type [RFC3564].

<Priority> = <Reservation Priority> <Preemption Priority>
<Defending Priority>

<Reservation priority> is an essential way to differentiate flows for emergency services, ETS, E911, etc., and assign them a higher admission priority than normal priority flows and best-effort priority flows. <Preemption Priority> is the priority of the new flow compared with the defending priority of previously admitted flows. Once a flow was admitted, the preemption priority becomes irrelevant. <Defending Priority> is used to compare with the preemption priority of new flows. For any specific flow, its preemption priority MUST always be less than or equal to the defending priority.

Appropriate security measures need to be taken to prevent abuse of the <Priority> parameters.

5.2.2 <QoS Available>

<QoS Available> = <Traffic Description> <Path Latency> <Path Jitter>
<Path BER> <Ctot> <Dtot> <Csum> <Dsum> <Priority>

<Path Latency>, <Path Jitter>, <Path BER>, <Ctot>, <Dtot>, <Csum>, and <Dsum> are optional QSPEC parameters.

These parameters describe the resources currently available on the path and hence the object is read-write. Each QNE MUST inspect this object, and if resources available to this QNE are less than what <QoS Available> says currently, the QNE MUST adapt it accordingly.

Hence when the message arrives at the recipient of the message, <QoS Available> reflects the bottleneck of the resources currently available on a path. It can be used in a QUERY message, for example, to collect the available resources along a data path. The <QoS Available> parameters are defined in [RFC 2210, 2212, 2215].

The <Traffic Description> parameters provide information, for example, about the bandwidth available along the path followed by a data flow. The local parameter is an estimate of the bandwidth the QNE has available for packets following the path. Computation of the value of this parameter SHOULD take into account all information available to the QNE about the path, taking into consideration administrative and policy controls on bandwidth, as well as physical resources. The composition rule for this parameter is the MIN function. The composed value is the minimum of the QNE's value and the previously composed value. This quantity, when composed end-to-end, informs the QNR (or QNI in a RESPONSE message) of the minimal bandwidth link along the path from QNI to QNR.

The <Path Latency> parameter accumulates the latency of the packet forwarding process associated with each QNE, where the latency is defined to be the smallest possible packet delay added by each QNE. This delay results from speed-of-light propagation delay, from packet processing limitations, or both. It does not include any variable queuing delay which may be present. Each QNE MUST add the propagation delay of its outgoing link, which includes the QNR adding

the associated delay for the egress link. Furthermore, the QNI MUST add the propagation delay of the ingress link. The composition rule for the <Path Latency> parameter is summation with a clamp of (2**32 - 1) on the maximum value. This quantity, when composed end-to-end, informs the QNR (or QNI in a RESPONSE message) of the minimal packet delay along the path from QNI to QNR. The purpose of this parameter is to provide a minimum path latency for use with services which provide estimates or bounds on additional path delay [RFC 2212]. Together with the queuing delay bound, this parameter gives the application knowledge of both the minimum and maximum packet delivery delay. Knowing both the minimum and maximum latency experienced by data packets allows the receiving application to know the bound on delay variation and de-jitter buffer requirements.

The <Path Jitter> parameter accumulates the jitter of the packet forwarding process associated with each QNE, where the jitter is defined to be the nominal jitter added by each QNE. IP packet

jitter, or delay variation, is defined in RFC 3393 [RFC3393], Section 4.6 (Type-P-One-way-peak-to-peak-ipdv), where the suggested evaluation interval is 1 minute. Note that the method to estimate peak-to-peak IP delay variation without active measurements requires more study. This jitter results from packet processing limitations, and includes any variable queuing delay which may be present. Each QNE MUST add the jitter of its outgoing link, which includes the QNR adding the associated jitter for the egress link. Furthermore, the QNI MUST add the jitter of the ingress link. The composition rule for the <Path Jitter> parameter is summation of a large percentage of the peak-to-peak variation with a clamp on the maximum value (note that the methods of accumulation and estimation of nominal QNE jitter is under study). This quantity, when composed end-to-end, informs the QNR (or QNI in a RESPONSE message) of the nominal packet jitter along the path from QNI to QNR. The purpose of this parameter is to provide a nominal path jitter for use with services that provide estimates or bounds on additional path delay [RFC 2212]. Together with the <Path Latency> and the queuing delay bound, this parameter gives the application knowledge of the typical packet delivery delay variation.

The <Path BER> parameter accumulates the bit error rate (BER) of the packet forwarding process associated with each QNE, where the BER is defined to be the smallest possible BER added by each QNE. Each QNE MUST add the BER of its outgoing link, which includes the QNR adding the associated BER for the egress link. Furthermore, the QNI MUST add the BER of the ingress link. The composition rule for the <Path BER> parameter is summation with a clamp on the maximum value (this assumes sufficiently low BER values such that summation error is not significant). This quantity, when composed end-to-end, informs the QNR (or QNI in a RESPONSE message) of the minimal packet BER along the path from QNI to QNR. As with <Jitter>, the method to estimate <Path BER> requires more study.

<Ctot>, <Dtot>, <Csum>, <Dsum>: Error terms C and D represent how the element's implementation of the guaranteed service deviates from the fluid model. These two parameters have an additive composition rule. The error term C is the rate-dependent error term. It represents the delay a datagram in the flow might experience due to the rate

parameters of the flow. The error term D is the rate-independent, per-element error term and represents the worst case non-rate-based transit time variation through the service element. If the composition function is applied along the entire path to compute the

end-to-end sums of C and D (<Ctot> and <Dtot>) and the resulting values are then provided to the QNR (or QNI in a RESPONSE message). <Csum> and <Dsum> are the sums of the parameters C and D between the last reshaping point and the current reshaping point.

5.2.3 <QoS Reserved>

<QoS Reserved> = <Traffic Description> <QoS Class> <Priority> <S>

These parameters describe the QoS reserved by the QNEs down the path, and hence the object is read-write.

<Traffic Description>, <QoS Class> and <Priority> are defined above.

<S> = slack term: difference between desired delay and delay obtained by using bandwidth reservation <Bandwidth> (used to reduce resource reservation for flow) [RFC 2212].

5.2.4 <Minimum QoS>

<Minimum QoS> = <Traffic Description> <QoS Class> <Priority>

<Minimum QoS> does not have an equivalent in RSVP. It allows the QNI to define a range of acceptable QoS levels by including both the desired QoS value and the minimum acceptable QoS in the same message. It is a read-only object. The desired QoS is included with a <QoS Desired> and/or a <QoS Available> object seeded to the desired QoS value. The minimum acceptable QoS value MAY be coded in the <Minimum QoS> object. As the message travels towards the QNR, <QoS Available> is updated by QNEs on the path. If its value drops below the value of <Minimum QoS> the reservation failed and can be aborted. When this method is employed the QNR SHOULD signal back to the QNI the value <QoS Available> attained in the end, because the reservation MAY need to be adapted accordingly.

Future versions of this document will describe how <Minimum QoS> can be used by the QNI to send a discrete set of desired parameters.

6. QSPEC Functional Specification

This Section defines the encodings of the QSPEC parameters and QSPEC control information defined in Section 4. We first give the general QSPEC formats and then the formats of the QSPEC objects.

6.1 General QSPEC Formats:

Note: This section is in a first draft state and further work is needed to define exact formats of objects.

Type: QSPEC
Length: Variable
Value: This object contains a 2 byte QOSM ID and variable length
       QSPEC information, which is QOSM specific.

```
  0                   1                   2                   3
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |              QOSM ID          |             Length            |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |   Object ID   | Parameter ID  |             Length            |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 //                      Parameter Values                       //
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |   Object ID   | Parameter ID  |             Length            |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 //                      Parameter Values                       //
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
                               ....
```

Object ID:
0: control information
1: QoS Desired
2: QoS Available
3: QoS Reserved
4: Min QoS

6.2 <NON NSLP Hop> & <NSLP Hops> Parameters [RFC 2210, 2215]

```
  0                   1
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |  NON NSLP Hop |    NSLP Hops  |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

NON NSLP Hop: 8 bits
    This field is set to 1 if a non NSLP-aware QNE is encountered on
    the path from the QNI to the QNR.

NSLP Hops: 8 bits
    Indicates the number of NSLP hops between the QNI and QNR.
    Values of the composed parameter will range from 1 to 255,
    limited by the bound specified by the <Max NSLP Hops> parameter.

6.3 <Max NSLP Hops> Parameter

```
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
| Max NSLP Hops |
+-+-+-+-+-+-+-+-+
```

Max NSLP Hops: 8 bits
Indicates the maximum number of NSLP hops between the QNI and QNR. Values of the composed parameter will range from 1 to 255, limited by the bound on IP hop count. The default value of <Max NSLP hops> is 255.

6.4 <Excess Treatment> Parameter

```
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
|     Excess    |
|   Treatment   |
+-+-+-+-+-+-+-+-+
```

Excess Treatment: 8 bits
Indicates how the QNE SHOULD process out-of-profile traffic.
Allowed values are as follows:
0: drop
1: shape
2: remark
The excess treatment parameter is initially set by the QNI and adjusted as needed by QNEs.

6.5 <Bandwidth> & <S> Parameters [RFC 2212, RFC 2215]

The <Bandwidth> parameter MUST be nonnegative and is measured in bytes per second and has the same range and suggested representation as the bucket and peak rates of the <Token Bucket>. <Bandwidth> can be represented using single-precision IEEE floating point. The representation MUST be able to express values ranging from 1 byte per second to 40 terabytes per second. For values of this parameter only valid non-negative floating point numbers are allowed. Negative numbers (including "negative zero"), infinities, and NAN's are not allowed.

A QNE MAY export a local value of zero for this parameter. A network element or application receiving a composed value of zero for this parameter MUST assume that the actual bandwidth available is unknown.

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Bandwidth       (32-bit IEEE floating point number)         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Slack Term [S]  (32-bit integer)                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Slack term S MUST be nonnegative and is measured in microseconds. The Slack term, S, can be represented as a 32-bit integer. Its value can range from 0 to (2**32)-1 microseconds.

6.6 <Token Bucket> Parameters [RFC 2215]

The <Token Bucket> parameters are represented by three floating Point numbers in single-precision IEEE floating point format followed by two 32-bit integers in network byte order. The first floating point value is the rate (r), the second floating point value is the bucket size (b), the third floating point is the peak rate (p), the first integer is the minimum policed unit (m), and the second integer is the maximum datagram size (M).


```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Token Bucket Rate [r] (32-bit IEEE floating point number)    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Token Bucket Size [b] (32-bit IEEE floating point number)    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Peak Data Rate [p] (32-bit IEEE floating point number)       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Minimum Policed Unit [m] (32-bit integer)                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Maximum Packet Size [M]  (32-bit integer)                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

When r, b, p, and R terms are represented as IEEE floating point values, the sign bit MUST be zero (all values MUST be non-negative). Exponents less than 127 (i.e., 0) are prohibited. Exponents greater than 162 (i.e., positive 35) are discouraged, except for specifying a peak rate of infinity. Infinity is represented with an exponent of all ones (255) and a sign bit and mantissa of all zeroes.

6.7 <QoS Class> Parameters

6.7.1 <PHB Class> Parameter [RFC 3170]

As prescribed in RFC 3170, the encoding for a single PHB is the recommended DSCP value for that PHB, left-justified in the 16 bit field, with bits 6 through 15 set to zero.

```
  0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|         DSCP          | 0   0   0   0   0   0   0   0   0   0 |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
```

The registries needed to use RFC 3140 already exist, see [DSCP-REGISTRY, PHBID-CODES-REGISTRY]. Hence, no new registry needs to be created for this purpose.

6.7.2 <Y.1541 QoS Class> Parameter [Y.1541]

Y.1541 QoS classes are defined as follows:

```
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
|    Y.1541     |
|  QoS Class    |
+-+-+-+-+-+-+-+-+
```

Y.1541 QoS Class: 8 bits
    Indicates the Y.1541 QoS Class. Values currently allowed are 0, 1, 2, 3, 4, 5.

Class 0:
Mean delay <= 100 ms, delay variation <= 50 ms, loss ratio <= 10-3. Real-time, highly interactive applications, sensitive to jitter. Application examples include VoIP, Video Teleconference.

Class 1:
Mean delay <= 400 ms, delay variation <= 50 ms, loss ratio <= 10-3. Real-time, interactive applications, sensitive to jitter. Application examples include VoIP, Video Teleconference.

Class 2:
Mean delay <= 100 ms, delay variation unspecified, loss ratio <= 10-3. Highly interactive transaction data. Application examples include signaling.

Class 3:
Mean delay <= 400 ms, delay variation unspecified, loss ratio <= 10-3. Interactive transaction data. Application examples include signaling.

Class 4:
Mean delay <= 1 sec, delay variation unspecified, loss ratio <= 10-3. Low Loss Only applications. Application examples include short transactions, bulk data, video streaming.

Class 5:
Mean delay unspecified, delay variation unspecified, loss ratio unspecified. Unspecified applications. Application examples include traditional applications of default IP networks.

6.7.3 <DSTE Class Type> Parameter [RFC3564]

DSTE class type is defined as follows:

```
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
|     DSTE      |
|  Class Type   |
+-+-+-+-+-+-+-+-+
```

DSTE Class Type: 8 bits
Indicates the DSTE class type. Values currently allowed are 0, 1, 2, 3, 4, 5, 6, 7.

6.8 Priority Parameters

6.8.1 <Preemption Priority> & <Defending Priority> Parameters [RFC 3181]

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Preemption Priority        |       Defending Priority      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Preemption Priority: 16 bits (unsigned)
The priority of the new flow compared with the defending priority of previously admitted flows. Higher values represent higher priority.

Defending Priority: 16 bits (unsigned)

6.8.2 <Reservation Priority> Parameter [SIP-PRIORITY]

```
 0                   1
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Reservation  |  Reservation  |
|   Priority    |   Priority    |
|   Namespace   |               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

High priority flows, normal priority flows, and best-effort priority flows can have access to resources depending on their {"Namespace", "Reservation Priority"} combination as follows:

Reservation Priority Namespace: 8 bits

- 0 - dsn high priority
- 1 - drsn high priority
- 2 - q735 high priority
- 3 - ets high priority
- 4 - wps high priority
- 5 - normal priority
- 6 - best-effort priority

Reservation Priority: 8 bits
Each namespace has a finite list of relative priority-values. Each is listed here in the order of lowest priority to highest priority:

4 - dsn.routine
3 - dsn.priority
2 - dsn.immediate
1 - dsn.flash
0 - dsn.flash-override

5 - drsn.routine
4 - drsn.priority
3 - drsn.immediate
2 - drsn.flash
1 - drsn.flash-override
0 - drsn.flash-override-override

4 - q735.4
3 - q735.3
2 - q735.2
1 - q735.1
0 - q735.0

4 - ets.4

3 - ets.3
2 - ets.2
1 - ets.1
0 - ets.0

4 - wps.4
3 - wps.3
2 - wps.2
1 - wps.1
0 - wps.0

0 - normal.0

0 - best.effort.0

Note that additional work is needed to communicate these flow priority values to bearer-level network elements [VERTICAL-INTERFACE].

6.9 <QoS Available> Parameters

6.9.1 <Path Latency> Parameter [RFC 2210, 2215]

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Path Latency (32-bit integer)                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The composition rule for the <Path Latency> parameter is summation with a clamp of (2**32 - 1) on the maximum value. The latencies are reported in units of one microsecond. An individual QNE can advertise a latency value between 1 and 2**28 (somewhat over two minutes) and the total latency added across all QNEs can range as high as (2**32)-2. If the sum of the different elements delays exceeds (2**32)-2, the end-to-end advertised delay SHOULD be reported as indeterminate. The distinguished value (2**32)-1 is taken to mean indeterminate latency. A QNE that cannot accurately predict the latency of packets it is processing SHOULD set its local parameter to this value. Because the composition function limits the composed sum to this value, receipt of this value at a network element or application indicates that the true path latency is not known. This MAY happen because one or more network elements could not supply a

value, or because the range of the composition calculation was exceeded.

6.9.2 <Path Jitter> Parameter

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Path Jitter (32-bit integer)                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The composition rule for the <Path Jitter> parameter is summation with a clamp of (2**32 - 1) on the maximum value. The jitters are reported in units of one microsecond. An individual QNE can advertise a jitter value between 1 and 2**28 (somewhat over two minutes) and the total jitter added across all QNEs can range as high as (2**32)-2. If the sum of the different elements delays exceeds (2**32)-2, the end-to-end advertised jitter SHOULD be reported as

indeterminate. The distinguished value (2**32)-1 is taken to mean indeterminate jitter. A QNE which cannot accurately predict the jitter of packets it is processing SHOULD set its local parameter to this value. Because the composition function limits the composed sum to this value, receipt of this value at a network element or application indicates that the true path jitter is not known. This MAY happen because one or more network elements could not supply a value, or because the range of the composition calculation was exceeded.

6.9.3 <Path BER> Parameter

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Path Bit Error Rate (32-bit integer)            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The composition rule for the <Path BER> parameter is summation with a clamp of 10^-2 on the maximum value. The BERs are reported in units of 10^-11. An individual QNE can advertise a BER value between 1 and 2**28 and the total BER added across all QNEs can range as high as (2**32)-2. If the sum of the different elements delays exceeds (2**32)-2, the end-to-end advertised BER SHOULD be reported as indeterminate. The distinguished value (2**32)-1 is taken to mean

indeterminate BER. A QNE which cannot accurately predict the BER of packets it is processing SHOULD set its local parameter to this value. Because the composition function limits the composed sum to this value, receipt of this value at a network element or application indicates that the true path BER is not known. This MAY happen because one or more network elements could not supply a value, or because the range of the composition calculation was exceeded.

6.9.4 <Ctot> <Dtot> <Csum> <Dsum> Parameters [RFC 2210, 2212, 2215]

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   End-to-end composed value for C [Ctot] (32-bit integer)     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   End-to-end composed value for D [Dtot] (32-bit integer)     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Since-last-reshaping point composed C [Csum] (32-bit integer) |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Since-last-reshaping point composed D [Dsum] (32-bit integer) |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The error term C is measured in units of bytes. An individual QNE can advertise a C value between 1 and 2**28 (a little over 250 megabytes) and the total added over all QNEs can range as high as (2**32)-1. Should the sum of the different QNEs delay exceed (2**32)-1, the end-to-end error term MUST be set to (2**32)-1. The error term D is measured in units of one microsecond. An individual QNE can advertise a delay value between 1 and 2**28 (somewhat over two minutes) and the total delay added over all QNEs can range as high as (2**32)-1. Should the sum of the different QNEs delay exceed (2**32)-1, the end-to-end delay MUST be set to (2**32)-1.

7. Security Considerations

The priority parameter raises possibilities for Theft of Service Attacks because users could claim an emergency priority for their flows without real need, thereby effectively preventing serious emergency calls to get through. Several options exist for countering such attacks, for example

- only some user groups (e.g. the police) are authorized to set the emergency priority bit

- any user is authorized to employ the emergency priority bit for particular destination addresses (e.g. police)

8. IANA Considerations

This section provides guidance to the Internet Assigned Numbers Authority (IANA) regarding registration of values related to the QSPEC template, in accordance with BCP 26 RFC 2434 [RFC2434].

[QoS-SIG] requires IANA to create a new registry for QoS Signaling Policy Identifiers. The QoS Signaling Policy Identifier (QOSM ID) is a 32 bit value carried in a QSPEC object. The allocation policy for new QOSM IDs is TBD.

This document also defines 24 objects for the QSPEC Template, as Detailed in Section 5. Values are to be assigned for them from the GIMPS Object Type registry.

9. Acknowledgements

The authors would like to thank David Black, Robert Hancock, Dave Oran, Tom Phelan, Hannes Tschofenig, and Sven van den Bosch, for their very helpful suggestions.

10. Intellectual Property Considerations

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any

copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

11. Normative References

[DSCP-REGISTRY] http://www.iana.org/assignments/dscp-registry
[PHBID-CODES-REGISTRY] http://www.iana.org/assignments/phbid-codes
[QoS-SIG] S. Van den Bosch et. al., "NSLP for Quality-of-Service Signaling," work in progress.
[RFC2119] S. Bradner, "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
[RFC2205] B. Braden et. al., "Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification," RFC 2205, September 1997.
[RFC2210] J. Wroclawski, "The Use of RSVP with IETF Integrated Services", RFC 2210, September 1997.
[RFC2211] J. Wroclawski, "Specification of the Controlled-Load Network Element Service", RFC 2211, Sept. 1997.
[RFC2212} Shenker, S., et. al., "Specification of Guaranteed Quality of Service," September 1997.
[RFC2215] S. Shenker and J. Wroclawski, "General Characterization Parameters for Integrated Service Network Elements", RFC 2215, Sept. 1997.
[RFC2475] S. Blake et. al., "An Architecture for Differentiated Services", RFC 2475, December 1998.

12. Informative References

[CMSS] "PacketCable (TM) CMS to CMS Signaling Specification, PKT-SP-CMSS-103-040402, April 2004.
[DIFFSERV-CLASS] Baker, F., et. al., "Configuration Guidelines for DiffServ Service Classes," work in progress.
[INTSERV-QOSM] C. Kappler, "A QoS Model for Signaling IntServ Controlled-Load Service with NSIS," work in progress.
[Q.2630] ITU-T Recommendation Q.2630.3: "AAL Type 2 Signaling Protocol - Capability Set 3" Sep. 2003
[RFC1633] B. Braden et. al., "Integrated Services in the Internet Architecture: an Overview," RFC 1633, June 1994.
[RFC3393] C. Demichelis, P. Chimento, "IP Packet Delay Variation Metric for IP Performance Metrics (IPPM), RFC 3393, November 2002.
[RFC3564] F. Le Faucheur et. al., Requirements for Support of Differentiated Services-aware MPLS Traffic Engineering, RFC 3564, July 2003
[RFC3726] M. Brunner et. al., "Requirements for Signaling Protocols", RFC 3726, April 2004.
[RMD-QOSM] A. Bader, L. Westberg, G. Karagiannis, C. Kappler and T. Phelan, " RMD-QOSM: An NSIS QoS Signaling Policy Model for Networks Using Resource Management in Diffserv (RMD)," work in progress.
[SIP-PRIORITY] H. Schulzrinne, J. Polk, "Communications Resource

Priority for the Session Initiation Protocol(SIP)." work in progress.
[VERTICAL-INTERFACE] M. Dolly, P. S. Tarapore, S. Sayers, "Discussion on Associating of Control Signaling Messages with Media Priority Levels," T1S1.7 & PRQC, October 2004.

[Y.1541] ITU-T Recommendation Y.1541, "Network Performance Objectives for IP-Based Services," May 2002.
[Y.1541-QOSM] J. Ash, et. al., "NSIS Network Service Layer Protocol QoS Signaling Proof-of-Concept," work in progress.

13. Authors' Addresses

Jerry Ash
AT&T
Room MT D5-2A01
200 Laurel Avenue
Middletown, NJ 07748, USA
Phone: +1-(732)-420-4578
Fax: +1-(732)-368-8659
EMail: gash@att.com

Attila Bader
Traffic Lab
Ericsson Research
Ericsson Hungary Ltd.
Laborc u. 1 H-1037
Budapest Hungary
EMail: Attila.Bader@ericsson.com

Chuck Dvorak
AT&T
Room 2A37
180 Park Avenue, Building 2
Florham Park, NJ 07932
Phone: + 1 973-236-6700
Fax:+1 973-236-7453
E-mail: cdvorak@att.com

Yacine El Mghazli
Alcatel
Route de Nozay
91460 Marcoussis cedex
FRANCE

Phone: +33 1 69 63 41 87
EMail: yacine.el_mghazli@alcatel.fr

Cornelia Kappler
Siemens AG
Siemensdamm 62
Berlin 13627
Germany
EMail: cornelia.kappler@siemens.com

Georgios Karagiannis
University of Twente
P.O. BOX 217
7500 AE Enschede
The Netherlands
EMail: g.karagiannis@ewi.utwente.nl

Andrew McDonald
Siemens/Roke Manor Research
Roke Manor Research Ltd.
Romsey, Hants SO51 0ZN
UK
EMail: andrew.mcdonald@roke.co.uk

Al Morton
AT&T
Room D3-3C06
200 S. Laurel Avenue
Middletown, NJ 07748
Phone: + 1 732 420-1571
Fax: +.1 732 368-1192
E-mail: acmorton@att.com

Percy Tarapore
AT&T
Room D1-33
200 S. Laurel Avenue
Middletown, NJ 07748
Phone: + 1 732 420-4172
E-mail: tarapore@.att.com

Lars Westberg
Ericsson Research

Torshamnsgatan 23
SE-164 80 Stockholm, Sweden
EMail: Lars.Westberg@ericsson.com

Appendix A: Example QSPECs

Note the mere definition of QSPECs is not sufficient for determining how to signal for DiffServ and IntServ respectively. Rather, the full QOSM needs to be defined.

A.1 QSPEC for Admission Control for DiffServ

QSPEC for Diffserv QOSM in general may be provided in future versions of this draft. A QSPEC for a DiffServ QOSM is included in [RMD-QOSM].

A.2 QSPEC for IntServ Controlled Load Service

The QoS Model for IntServ Controlled Load is defined in [RFC2211]. The QSPEC can be derived from usage of RSVP to signal for this QoS Model, as defined in [RFC2210] and [RFC2215].

The QSPEC for IntServ Controlled Load is composed of the objects <QoS Desired> and <QoS Available>, as well as <QoS Reserved>. Which object is present in a particular QSPEC depends on the message type (RESERVE, QUERY etc) in which the QSPEC travels. Details MUST be provided in a corresponding QOSM. Parameters in the QSPEC are as follows:


```
<QSPEC Control Information> = <NON NSLP Hop> <NSLP Hops>
<QoS Description> = <QoS Desired> <QoS Available> <QoS Reserved>
<QoS Desired> = <Token Bucket>
<QoS Available> = <Bandwidth> <Path Latency> <M>
<QoS Reserved> = <Token Bucket>
```

An IntServ over Diffserv QSPEC is

```
<QSPEC Control Information> = <NON NSLP Hop> <NSLP Hops>
<QoS Desired> = <Token Bucket>
<QoS Class> = <PHB Class>
<QoS Available> = <Bandwidth> <Path Latency> <M>
<QoS Reserved> = <Token Bucket>
```

Or a simple QSPEC for Diffserv requesting bandwidths for different PHBs is

```
<QoS Desired> = <Bandwidth>
<QoS class> = <PHB Class>
```

A.3 QSPEC for IntServ Guaranteed Services

The QoS Model is defined in [RFC 2212]. The required parameters to achieve guarantied service with RSVP are specified in [RFC 2210] and [RFC 2215].

The QSPEC for guaranteed services is the following:

```
<QSPEC Control Information> = <NON NSLP Hop> <NSLP Hops>
<QoS Description> = <QoS Desired> <QoS Available> <QoS Reserved>
<QoS Desired> = <Token Bucket>
```

This object contains token bucket parameters defined in [RFC 2210]. (equivalent to TSpec defined in RSVP).

```
<QoS Available> = <Bandwidth> <Path Latency> <MTU> <Ctot> <Dtot>
                  <Csum> <Dsum>
```

These parameters are defined in [RFC 2212] and [RFC 2215]. This object is equivalent to the AdSpec of RSVP.

```
<QoS Reserved> = <Token Bucket> <Bandwidth> <S>
```
Requested Rate and Slack Term defined in [RFC 2212], equivalent to RSpec of RSVP.

Note that the Guaranteed Services QoS Model only works with receiver initiated reservation signaling, because <Bandwidth> and <S> are derived from parameters contained in <QoS Available>, and MAY be updated on the return paths.

Appendix B: QoS Models and QSPECs

This Appendix gives a description of QoS Models and QSPECs and explains what is the relation between them. Once these descriptions are contained in a stable form in the appropriate IDs this Appendix

will be removed.

QoS NSLP is a generic QoS signaling protocol that can signal for many QOSMs. A QOSM is a particular QoS provisioning method or QoS architecture such as IntServ Controlled Load or Guaranteed Service, DiffServ, or RMD for DiffServ.

The definition of the QOSM is independent from the definition of QoS NSLP. Existing QOSMs do not specify how to use QoS NSLP to signal for them. Therefore, we need to define the QOSM specific signaling functions, as [RMD-QOSM], [INTSERV-QOSM], and [Y.1541-QOSM].

A QOSM SHOULD include the following information:

- Role of QNEs in this QOSM:
E.g. location, frequency, statefulness...
- QSPEC Definition:
A QOSM SHOULD specify the QSPEC, including QSPEC parameters. Furthermore it needs to explain how QSPEC parameters not used in this QOSM are mapped onto parameters defined therein.
- Message Format
Objects to be carried in RESERVE, QUERY RESPONSE and NOTIFY
- State Management
It describes how QSPEC info is treated and interpreted in the RMF and QOSM specific processing. E.g. admission control, scheduling, policy control, QoS parameter accumulation (e.g. delay).
- Operation and Sequence of Events
Usage of QoS-NSLP messages to signal the QOSM.

Appendix C: Mapping of QoS Desired, QoS Available and QoS Reserved of NSIS onto AdSpec, TSpec and RSpec of RSVP IntServ

The union of QoS Desired, QoS Available and QoS Reserved can provide all functionality of the objects specified in RSVP IntServ, however it is difficult to provide an exact mapping.

In RSVP, the Sender TSpec specifies the traffic an application is going to send (e.g. token bucket). The AdSpec can collect path characteristics (e.g. delay). Both are issued by the sender. The receiver sends the FlowSpec which includes a Receiver TSpec describing the resources reserved using the same parameters as the Sender TSpec, as well as a RSpec which provides additional IntServ QoS Model specific parameters, e.g. Rate and Slack.

The RSVP TSpec/AdSpec/RSpec seem quite tailored to receiver-initiated signaling employed by RSVP, and the IntServ QoS Model. E.g. to the knowledge of the authors it is not possible for the sender to specify a desired maximum delay except implicitly and mutably by seeding the AdSpec accordingly. Likewise, the RSpec is only meaningfully sent in the receiver-issued RSVP RESERVE message. For this reason our discussion at this point leads us to a slightly different mapping of necessary functionality to objects, which should result in more

flexible signaling models.

Full Copyright Statement

Disclaimer of Validity