INTERNATIONAL TELECOMMUNICATION UNION

ITU-T J.163

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

(03/2004)

SERIES J: CABLE NETWORKS AND TRANSMISSION OF TELEVISION, SOUND PROGRAMME AND OTHER MULTIMEDIA SIGNALS

IPCablecom

Dynamic quality of service for the provision of real-time services over cable television networks using cable modems

ITU-T Recommendation J.163

ITU-T Recommendation J.163

Dynamic quality of service for the provision of real-time services over cable television networks using cable modems

Summary

Many cable television operators are upgrading their facilities to provide two-way capability and using this capability to provide high-speed IP data services per ITU-T Recs J.83 and J.112. These operators now want to expand the capability of this delivery platform to include telephony. This Recommendation is one of a series of Recommendations required to achieve this goal. It provides for the dynamic quality of service needed in many real-time applications.

This Recommendation is revised as to comply with a number of relevant industry developments since ITU-T Rec. J.163 was released and in conformity with the current version of the CableLabs PacketCable™ specification.

Due to a change in the technological landscape within Europe, Annex A is removed from the Recommendation as it is no longer needed. As part of this change, Annex B is folded into the main body of the Recommendation and the term "AN" (Access Node) is replaced by "CMTS" (Cable Modem Termination System) throughout.

Source

ITU-T Recommendation J.163 was approved on 15 March 2004 by ITU-T Study Group 9 (2001-2004) under the ITU-T Recommendation A.8 procedure.

FOREWORD

The International Telecommunication Union (ITU) is the United Nations specialized agency in the field of telecommunications. The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of ITU. ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Assembly (WTSA), which meets every four years, establishes the topics for study by the ITU-T study groups which, in turn, produce Recommendations on these topics.

The approval of ITU-T Recommendations is covered by the procedure laid down in WTSA Resolution 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

Compliance with this Recommendation is voluntary. However, the Recommendation may contain certain mandatory provisions (to ensure e.g., interoperability or applicability) and compliance with the Recommendation is achieved when all of these mandatory provisions are met. The words "shall" or some other obligatory language such as "must" and the negative equivalents are used to express requirements. The use of such words does not suggest that compliance with the Recommendation is required of any party.

INTELLECTUAL PROPERTY RIGHTS

ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, ITU had not received notice of intellectual property, protected by patents, which may be required to implement this Recommendation. However, implementors are cautioned that this may not represent the latest information and are therefore strongly urged to consult the TSB patent database.

CONTENTS

ITU-T Recommendation J.163

Dynamic quality of service for the provision of real-time services over cable television networks using cable modems

1 Scope

This Recommendation addresses requirements for a client device to obtain access to network resources. In particular, it specifies a comprehensive mechanism for a client device to request a specific Quality of Service from the J.112 network. Extensive examples illustrate the use of this Recommendation. The scope of this Recommendation is to define the QoS Architecture for the "Access" portion of the IPCablecom network, provided to requesting applications on a per-flow basis.

2 References

The following ITU-T Recommendations and other references contain provisions which, through reference in this text, constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and other references are subject to revision; users of this Recommendation are therefore encouraged to investigate the possibility of applying the most recent edition of the Recommendations and other references listed below. A list of the currently valid ITU-T Recommendations is regularly published. The reference to a document within this Recommendation does not give it, as a stand-alone document, the status of a Recommendation.

2.1 Normative references

- ITU-T Recommendation J.83 (1997), *Digital multi-programme systems for television, sound and data services for cable distribution*.
- ITU-T Recommendation J.112 (1998), *Transmission systems for interactive cable television services*.
- ITU-T Recommendation J.112 Annex A (2001), *Digital Video Broadcasting: DVB interaction channel for Cable TV (CATV) distribution systems*.
- ITU-T Recommendation J.112 Annex B (2004), *Data-over-cable service interface specifications: Radio-frequency interface specification*.
- ITU-T Recommendation J.160 (2002), *Architectural framework for the delivery of time-critical services over cable television networks using cable modems*.
- ITU-T Recommendation J.161 (2001), *Audio codec requirements for the provision of bidirectional audio service over cable television networks using cable modems*.
- IETF RFC 1321 (1992), *The MD5 Message-Digest Algorithm*.
- IETF RFC 2205 (1997), *Resource ReSerVation Protocol (RSVP) – Version 1 Functional Specification*. (Updated by RFC 2750.)
- IETF RFC 2210 (1997), *The Use of RSVP with IETF Integrated Services*.
- IETF RFC 2748 (2000), *The COPS (Common Open Policy Service) Protocol*.
- IETF RFC 2865 (2000), *Remote Authentication Dial In User Service (RADIUS)*.

2.2 Informative references

- ITU-T Recommendation G.114 (2003), *One-way transmission time*.
- ITU-T Recommendation G.711 (1988), *Pulse code modulation (PCM) of voice frequencies*.
- ITU-T Recommendation G.726 (1990), *40, 32, 24, 16 kbit/s adaptive differential pulse code modulation (ADPCM)*.
- ITU-T Recommendation G.728 (1992), *Coding of speech at 16 kbit/s using low-delay code excited linear prediction*.
- ITU-T Recommendation G.729 Annex E (1998), *11.8 kbit/s CS-ACELP speech coding algorithm*.
- ITU-T Recommendation J.162 (2004), *Network call signalling protocol for the delivery of time-critical services over cable television networks using cable modems*.
- ITU-T Recommendation J.164 (2001), *Event message requirements for the support of real-time services over cable television networks using cable modems*.
- ITU-T Recommendation J.170 (2002), *IPCablecom security specification*.
- IETF RFC 791 (1981), *Internet Protocol – DARPA Internet Program – Protocol specification*.
- IETF RFC 1890 (1996), *RTP Profile for Audio and Video Conferences with Minimal control*.
- IETF RFC 2327 (1998), *SDP: Session Description Protocol*.
- IETF RFC 2474 (1998), *Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers*.
- IETF RFC 2543 (1999), *SIP: Session Initiation Protocol*.
- IETF RFC 2749 (2000), *COPS usage for RSVP*.
- IETF RFC 2750 (2000), *RSVP Extensions for Policy Control*.
- IETF RFC 2753 (2000), *A Framework for Policy-based Admission Control*.
- IETF RFC 2866 (2000), *RADIUS Accounting*.
- IETF RFC 2961 (2001), *RSVP Refresh Overhead Reduction Extensions*.
- IETF RFC 2996 (2000), *Format of the RSVP DCLASS Object*.
- IETF RFC 3006 (2000), *Integrated Services in the Presence of Compressible Flows*.
- IETF RFC 3209 (2001), *RSVP-TE: Extensions to RSVP for LSP Tunnels*.
- IETF RFC 3084 (2001), *COPS Usage for Policy Provisioning (COPS-PR)*.
- *PacketCable Distributed Call Signalling Specification,* PKT-SP-DCS-D03-000428, 28 April 2000.
- *PacketCable Dynamic Quality-of-Service Specification,* PK-SP-DQOS-I07-03-08-15.

3 Terms and definitions

This Recommendation defines the following terms:

3.1 cable modem: A cable modem is a layer two termination device that terminates the customer end of the J.112 (or J.122) connection.

3.2 J.112 flow: A unidirectional or bidirectional flow of data packets that is subject to MAC-layer signalling and QoS assignment compliant to ITU-T Rec. J.112 (or ITU-T Rec. J.122).

3.3 IPCablecom: An ITU-T project that includes an architecture and a series of Recommendations that enable the delivery of real-time services over the cable television networks using cable modems.

4 Abbreviations and Conventions

4.1 Abbreviations

This Recommendation uses the following abbreviations:

CM	Cable Modem
CMTS	Cable Modem Termination System
COPS	Common Open Policy Service
CPE	Customer Premises Equipment
DCS	Distributed Call Signalling
DSA	Dynamic Service Addition
DSC	Dynamic Service Change
INA	Interactive Network Adapter
IP	Internet Protocol
MTA	Media Terminal Adaptor
NCS	Network-based Call Signalling
PHS	Payload Header Suppression
PSTN	Public Switched Telephone Network
QoS	Quality of Service
RAP	Resource Allocation Protocol
RSVP	Resource ReSerVation Protocol
TLV	Type-Length-Value
VAD	Voice Activity Detection

4.2 Conventions

Throughout this Recommendation, the words that are used to define the significance of particular requirements are capitalized. These words are:

"MUST"	This word or the adjective "REQUIRED" means that the item is an absolute requirement of this Recommendation.
"MUST NOT"	This phrase means that the item is an absolute prohibition of this Recommendation.

"SHOULD" This word or the adjective "RECOMMENDED" means that there may exist valid reasons in particular circumstances to ignore this item, but the full implications should be understood and the case carefully weighed before choosing a different course.

"SHOULD NOT" This phrase means that there may exist valid reasons in particular circumstances when the listed behaviour is acceptable or event useful, but the full implications should be understood and the case carefully weighed before implementing any behaviour described with this label.

"MAY" This word or the adjective "OPTIONAL" means that this item is truly optional. One vendor may choose to include the item because a particular marketplace requires it or because it enhances the product, for example; another vendor may omit the same item.

5 Technical overview

Enhanced Quality of Service is required for supporting interactive multimedia applications. Resources may be constrained in segments of the network, requiring allocation of resources in the network. The scope of this Recommendation is to define the Quality of Service Architecture for the "Access" portion of the IPCablecom network. The access portion of the network is defined to be between the Multimedia Terminal Adaptor (MTA) and the Cable Modem Termination System (CMTS), including the J.112 network. This Recommendation also recognizes that per-flow reservations may be required within the customer premises, and the protocols developed herein address this potential need. Although some segments of the backbone network may require resource reservation to provide adequate quality of service, we consider the protocols for backbone resource management to be outside the scope of this Recommendation.

Resources are allocated on the J.112 network for individual flows associated with each session of an application, per subscriber, on an authorized and authenticated basis. A DQoS session, or simply a session, is defined by this Recommendation to be a single bidirectional data flow between two clients. When a multimedia application needs multiple bidirectional data flows (e.g., one for voice and a separate for video), separate DQoS sessions are established for each. Applications may use only half of the session's bidirectional data flow, thereby providing send-only or receive-only services. For example, in a typical voice communications application, a simple communication between two parties is implemented by a single session, while complex, multiparty communications (e.g., "conference calls") are implemented by multiple simultaneous sessions.

Two IPCablecom Call Signalling protocols are being defined – Network-based Call Signalling (ITU-T Rec. J.162) and Distributed Call Signalling (IETF RFC 2543 SIP). This Dynamic QoS specification is the underlying QoS framework for both of these call signalling protocols. QoS is allocated for flows associated with a session in concert with the signalling protocol.

This Recommendation introduces the concept of a segment-by-segment QoS framework. It exploits the information available from signalling protocols to perform the QoS assignment on both the "local" segment (on the J.112 network close to the originating party) and the "remote" segment (the J.112 network close to the terminating party). Thus, this Recommendation allows different providers to use the most appropriate mechanisms for the segment that they are managing. Using a concatenation of the segments with QoS, we provide end-to-end QoS assurance for the session.

The Dynamic QoS specification incorporates protocols to enable providers of packet-based voice communications using the IPCablecom framework to use different charging models, including both flat-rate charging as well as usage-based charging. It is the intent of this Recommendation to ensure that enhanced QoS is provided only to authorized and authenticated users. The specific techniques used for authorizing and authenticating a user are beyond the scope of this Recommendation.

This Dynamic QoS specification recognizes the requirements of a commercially viable voice communications service analogous to that offered by means of the public switched telephone network. It is important to ensure that resources are available before the two parties involved in the session are invited to communicate. Thus, resources are reserved before the recipient of the communication is notified that someone is trying to initiate a communication. If there are insufficient resources for a session, then the session is blocked.

The protocols developed in this Recommendation explicitly recognize the need to ensure that there is no potential for fraud or theft of service by endpoints that do not wish to cooperate with the call signalling and QoS signalling protocols with the intent of avoiding being charged for usage. This Recommendation introduces the concept of a two-phase activation for resource reservation (reserve and commit). The two phases allow a provider to both allocate resources only when they are required (when the voice path is cut-through) which may be used for billing. Further, because the second phase to commit resources requires an explicit request from the MTA, it enables the provider to prevent fraud and theft of service.

This Recommendation is technically compatible with the corresponding CableLabs PacketCable document: *PacketCable Dynamic Quality-of-Service Specification* PK-SP-DQOS-I07-03-08-15.

5.1 IPCablecom QoS architecture requirements

The following list presents the QoS requirements for supporting multimedia applications over IPCablecom Networks.

1) *Provide IPCablecom accounting for the QoS resources on a per-session basis*

   It is anticipated that, from a billing perspective, one of the resources that will need to be accounted for is the use of QoS in the J.112 network. Thus, information needs to be identified and tracked that allows reconciliation of the use of the J.112 QoS resource with IPCablecom session activity.

2) *Both two-phase (reserve-commit) and single-phase (commit) QoS activation models*

   Under application control it should be possible to utilize either a two-phase or single-phase QoS activation model. In the two-phase model the application reserves the resource, then later commits it. In the single-phase model both reservation and commitment occur as a single autonomous operation. As in the J.112 model, resources that are reserved but not yet committed are available for temporary assignment to other (e.g., best effort) J.112 Flows. This Recommendation provides mechanisms for both two-phase and single-phase activation for embedded MTAs, and for two-phase activation for stand-alone MTAs. Single-phase activation for stand-alone MTAs is deferred to later releases of this Recommendation.

3) *Provide IPCablecom defined policies to control QoS in both the J.112 network and the IP backbone*

   It should be possible for different types of sessions to have different QoS characteristics. For example, sessions within a single CABLE OPERATOR provider's domain may receive different QoS than sessions outside the domain (e.g., international sessions including links to the PSTN). This dynamic QoS specification may allow a CABLE OPERATOR to provide different QoS for different types of customers (e.g., higher QoS for subscribers of a business service at certain times of the day compared to residential customers), or different types of applications for a single customer.

4) *Prevent (minimize) abusive QoS usage*

   Two types of abusive QoS usage are identified: that which is accurately billed but leads to denying service to others, and that which is not accurately billed and leads to theft of service. Subscriber applications and IPCablecom applications (either embedded or

PC-based) may inadvertently or intentionally abuse their QoS privileges (e.g., use of enhanced QoS, which the provider wants limited to voice applications, by an FTP application). Even though the J.112 network is expected to enforce a subscriber's access to QoS, rich packet classification and signalling control mechanisms should exist to keep the subscriber (and the subscriber devices) from fraudulent use of QoS. Admission control procedures should be employed to reduce denial-of-service attacks.

5) *Provide admission control mechanisms for both upstream and downstream directions in the J.112 network*

Both upstream and downstream QoS should be subject to per-session admission control.

6) *Use QoS mechanism of the J.112 MAC layer*

It should be possible to police (defined as marking, dropping, or delaying packets) all aspects of QoS defined in the service at the CMTS using the J.112 QoS mechanisms. Furthermore, it should be possible to support multiple flow mapping models – associate a single IPCablecom session to a single J.112 Flow and multiple IPCablecom sessions to a single J.112 Flow.

7) *Policy is enforced by the CMTS*

Ultimate policy control is entrusted to the CMTS. The philosophy is that any client can make any QoS request, but the CMTS (or an entity behind the CMTS) is the only entity entrusted to grant or deny QoS requests.

8) *IPCablecom entities must be as unaware as possible of specific J.112 QoS primitives and parameters*

For IPCablecom, like any other application that uses the IP-network, the design objective is to minimize the amount of access-link-specific knowledge contained within the application layer. The less access-link knowledge in the application layer, the more applications will be available for development and deployment, and the fewer testing and support-problems will be encountered.

9) *Reclamation of QoS resources for dead/stale sessions*

It is necessary to reclaim and reallocate precious QoS resources for sessions that are no longer active, but have not been properly torn down. There should be no resource "leaks" in the J.112 link. For example, if an IPCablecom client module malfunctions in the midst of an IPCablecom session, all J.112 QoS resources used by the session should be released within a reasonable period of time.

10) *Dynamic QoS policy changes*

It is desirable to dynamically change QoS policies for subscribers. For example, this requirement addresses the ability to change a customer's service level (e.g., upgraded from a "bronze" service to a "gold" service) on-the-fly without resetting the CM.

11) *Absolute minimum session set-up latency time and post pick-up delay*

The IPCablecom Network should allow for emulation and enhancement of the PSTN experience to the user, and should be equally good, if not better, in session set-up and post pick-up delay metrics.

12) *Multiple concurrent sessions*

It is desirable to allocate QoS resources (e.g., bandwidth) for not only individual point-to-point sessions, but also for multiple point-to-point sessions (e.g., conference calls, combined audio/video calls).

13) *Dynamic adjustment of QoS parameters in the middle of IPCablecom sessions*

It should be possible for the IPCablecom service to change QoS mid-session, e.g., network-wide resource adjustments or creation of compatible CODEC parameters (necessitating QoS changes), or user defined feature to vary QoS levels, or detection of fax or modem streams (necessitating change from compressing CODEC to ITU-T Rec. G.711).

14) *Support multiple QoS control models*

Strong cases can be made for both subscriber-side and network-side initiation of QoS signalling. In subscriber side signalling, an application can initiate its request for QoS immediately when the application believes it needs QoS. Also, subscriber side signalling supports application models that are peer-to-peer. In network-side signalling, implementation of the endpoint application can be completely unaware of QoS (especially in the J.112 network). Network-side signalling supports application models that are client-server (with the server being trusted). It is expected that both models will be present in IPCablecom (and other application) networks. This Recommendation is for subscriber-side signalling only.

15) *Support both embedded-MTA and stand-alone-MTA QoS signalling*

It should be possible to signal QoS from both an embedded-MTA and stand-alone-MTA. In a stand-alone MTA the only signalling path supported is that specified herein using RSVP. In an embedded MTA, both RSVP and direct access to the J.112 MAC signalling is possible.

5.2 IP QoS access network elements

The following network elements are employed to support QoS for IPCablecom Networks.

5.2.1 Multimedia Terminal Adaptor (MTA)

The IPCablecom network client device (i.e., the MTA) can be one of the following devices. These devices reside at the customer site and are connected through the J.112 channel to the network. All MTAs are assumed to implement some multimedia signalling protocol, such as J.162. An MTA may be either a device with a standard two-wire telephone set in the MTA-1 configuration, or may add video input/output capabilities in the MTA-2 configuration. It may have minimal capabilities, or may implement this functionality on a multimedia personal computer, and have all of the capabilities of the PC at its disposal.

From the point of view of QoS, there are two types of MTAs.

1) **Embedded/Integrated MTA**: This is a client multimedia terminal which incorporates a J.112 MAC-layer interface to the J.112 network.

2) **Stand-alone MTA**: This is a Client that implements the multimedia functionality without incorporating a J.112 MAC-layer interface. The stand-alone MTA will typically use Ethernet, USB, or IEEE 1394 as the physical interconnect to a CM. The stand-alone MTA may be connected to a customer network, and use transport facilities of the customer network (possibly including intermediate IP routers) to establish sessions over the J.112 network.

5.2.2 Cable Modem (CM)

This is an IPCablecom network element as defined by ITU-T Rec. J.112. The CM is responsible for classifying, policing and marking packets once the traffic flows are established by the signalling protocols described herein.

5.2.3 Cable Modem Termination System (CMTS)

The CMTS is the element of the IPCablecom network that contains centralized functions responsible for processing information flows. The CMTS acts as a Policy Enforcement Point (PEP) per the IETF Resource Allocation Protocol (RAP) Framework.

The CMTS implements a "IPCablecom Dynamic QoS Gate" (hereafter called just "Gate") between the J.112 network and an IP Backbone. The Gate is implemented using the packet classification and filtering functions defined in ITU-T Rec. J.112.

The CMTS may or may not also be configured as an "IS-DS Boundary" entity. An IS-DS Boundary interfaces to an internetwork using the Integrated Services (Intserv) model of QoS control and some other model, e.g., Differentiated Services (Diffserv).

5.2.4 Call Management Server (CMS) and Gate Controller (GC)

The IPCablecom Call Management Server (CMS) entity performs services that permit MTAs to establish Multimedia sessions (including voice communications applications such as "IP telephony" or "VoIP"). A CMS using the Network-Controlled call signalling model implements a Call Agent that directly controls the session, and maintains per-call state. A CMS using the Distributed Call signalling model may serve as a "DCS Proxy" and perform services only during initial session set-up. The term Gate Controller (GC) is used to refer to the portion of either type of CMS that performs the Quality of Service related functions.

In the IPCablecom Dynamic QoS Model, the Gate Controller controls the operation of the Gates implemented on an CMTS. The GC acts as a Policy Decision Point (PDP) per the IETF Resource Allocation Protocol (RAP) Framework.

5.2.5 Record Keeping Server (RKS)

The Record Keeping Server is a IPCablecom network element that only receives information from IPCablecom elements described in this Recommendation. The RKS can be used as a billing server, diagnostic tool, etc.

5.3 IPCablecom dynamic QoS architecture

The IPCablecom QoS architecture is based upon ITU-T Rec. J.112, IETF RSVP, and IETF Integrated Services Guaranteed QoS.

Specifically, the IPCablecom QoS architecture uses the protocol as defined in ITU-T Rec. J.112 within the cable television network. These messages support static and dynamic installation of packet classifiers (i.e., Filter-Specs) and flow scheduling (i.e., flow specs) mechanisms to deliver enhanced quality of service. J.112 QoS is based upon the objects which describe traffic and flow specifications, similar to the TSPEC and RSPEC objects as defined in the IETF Resource reSerVation Protocol (RSVP). This allows QoS resource reservations to be defined on a per flow basis.

In the J.112 QoS architecture, J.112 Flows are considered to be either unidirectional or bidirectional. In each direction, J.112 Flows are subject to the operations shown below.

The CM, where traffic enters the QoS enabled J.112 network, is responsible for:

- Classification of IP traffic into J.112 Flows based on defined filter specifications.
- Performing traffic shaping and policing as required by the flow specification.
- Maintaining state for active flows.
- Altering the TOS field in the upstream IP headers based on the network operator's policy.
- Obtaining the required J.112 QoS from the CMTS.
- Applying J.112 QoS mechanisms appropriately.

The CMTS is responsible for:

- Providing the required QoS to the CM based upon policy configuration.
- Allocating upstream bandwidth in accordance with CM requests and network QoS policies.
- Classifying each arriving packet from the network side interface and assigning it to a QoS level based on defined filter specifications.
- Policing the TOS field in received packets from the J.112 network to enforce TOS field settings per network operator policy.
- Altering the TOS field in the downstream IP headers based on the network operator's policy.
- Performing traffic shaping and policing as required by the flow specification.
- Forwarding downstream packets to the J.112 network using the assigned QoS.
- Forwarding upstream packets to the backbone network devices using the assigned QoS.
- Maintaining state for active flows.

The backbone network may either utilize IETF Integrated Services based mechanisms or use IETF Differentiated Services mechanisms. In a Diffserv backbone, network routers forward a packet, providing the appropriate IETF QoS, based on the setting of the TOS field. In a Diffserv backbone, no per-flow state is required in the core network devices.

5.4 QoS interfaces

Quality of service signalling interfaces are defined between many of the components of the IPCablecom network as shown in Figure 1. Signalling involves communication of QoS requirements at the application layer (e.g., SDP parameters), network layer (e.g., RSVP), and at the data-link layer (e.g., J.112 QoS). Also, the requirement for policy enforcement and system linkages between the OSS subscriber provisioning, admission control within the managed IP backbone, and admission control within the J.112 network creates the need for additional interfaces between components in the IPCablecom network.

An expanded explanation of QoS architecture framework is contained in the IPCablecom Architecture Framework, ITU-T Rec. J.160, and is shown in Figure 1.

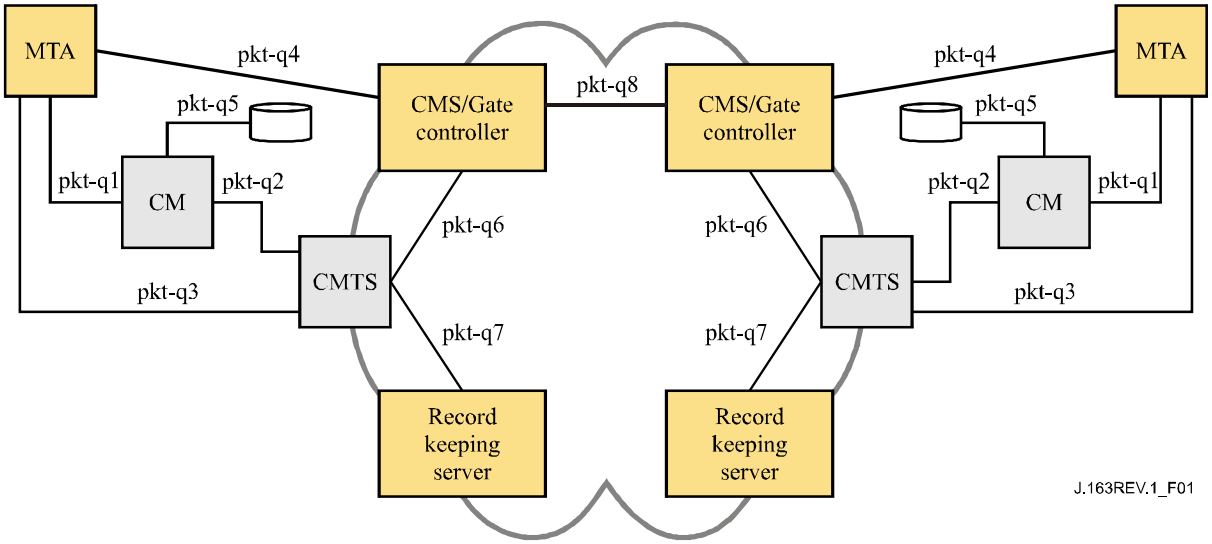


Figure 1/J.163 – QoS signalling interfaces in IPCablecom network

Interfaces pkt-q1 through pkt-q8 are available for controlling and processing QoS. Not all interfaces are used in all configurations and protocol variations. All but the pkt-q5 interface are utilized by DQoS. Table 1 briefly identifies each interface and how each interface is used in this Dynamic QoS Specification (DQoS). Two alternatives are shown for this specification: first a general interface that is applicable to either embedded or stand-alone MTAs; and second, an optional interface that is available only to embedded MTAs.

Table 1/J.163 – DQoS interfaces

Interface	Description	DQoS embedded/ Stand-alone MTA	DQoS embedded MTA (optional)
pkt-q1	MTA-CM	N/A	J.112 MAC-layer interface
pkt-q2	CM-CMTS	J.112 QoS, CMTS-initiated	J.112 QoS, CM-initiated
pkt-q3	MTA-CMTS	RSVP+	N/A
pkt-q4	MTA-GC/CMS	NCS/DCS	NCS/DCS
pkt-q5	CM-Provisioning Server	N/A	N/A
pkt-q6	GC-CMTS	Gate Management	Gate Management
pkt-q7	CMTS-RKS	Billing	Billing
pkt-q8	CMS-CMS	CMS-to-CMS Signalling	CMS-to-CMS Signalling

pkt-q1: Interface between the MTA and CM

This interface is only defined for the embedded MTA. The interface decomposes into three sub-interfaces:

- Control: used to manage J.112 Flows and their associated QoS traffic parameters and classification rules.
- Synchronization: used to synchronize packetization and scheduling for minimizing latency and jitter.
- Transport: used to process packets in the media stream and perform appropriate per-packet QoS processing.

This interface is conceptually defined in ITU-T Rec. J.112. For stand-alone MTAs, no instance of this interface is defined.

pkt-q2: J.112 QoS interface between CM and CMTS

This is the J.112 QoS interface (control, scheduling and transport). Control functions can be initiated from either the CM or the CMTS. However the CMTS is the final policy arbiter and granter of resources by performing admission control for the J.112 network. This interface is defined in ITU-T Rec. J.112.

pkt-q3: Network layer interface between the MTA and CMTS

The interface is used to request bandwidth, and QoS in terms of delay using standard RSVP and extensions specified herein. As a result of message exchanges between the MTA and CMTS, J.112 Flows are activated using CMTS-originated signalling on interface pkt-q2.

pkt-q4: Application layer signalling between GC/CMS and MTA

Many parameters are signalled across this interface such as the media stream, IP addresses, port numbers, and the selection of Codec and packetization characteristics. DCS and NCS are two examples of application layer signalling.

pkt-q5: Signalling from the J.112/IPCablecom provisioning to the CM

This interface is not utilized for QoS signalling in DQoS.

pkt-q6: Interface between the GC/CMS and CMTS

This interface is used to manage the dynamic Gates for media stream sessions. This interface enables the IPCablecom network to request and authorize QoS. With respect to admission and authorization, in the context of IPCablecom, a trust relationship must exist between the GC/CMS and CMTS.

pkt-q7: CMTS to Record Keeping Server

This interface is used by the CMTS to signal to the RKS all changes in session authorization and usage.

pkt-q8: CMS to CMS interface

This interface is used for session management and resource coordination between a pair of CMSs.

5.5 Framework for IPCablecom QoS

In order to justify its costs to the end user, a commercial multimedia service (e.g., voice communications capability) may require a high level of transport and signalling performance, including:

- Low delay: End-to-end packet delay needs to be small enough that it does not interfere with normal multimedia interactions. For normal telephony service using the PSTN, the ITU-T recommends no greater than 300 ms roundtrip delay[1]. Given that the end-to-end backbone propagation delay may absorb a significant amount of this delay budget, it is important to control delay on the access channel, at least for long-distance calls.
- Low packet loss: Packet loss needs to be small enough so that voice quality or performance of fax and voiceband modems is not perceptibly impaired. While loss concealment algorithms can be used to reproduce intelligible speech even with high loss rates, the resulting performance cannot be considered to be adequate as a replacement for existing circuit-switched telephone service. Loss requirements for acceptable voiceband modem performance are even more stringent than those for voice.
- Short post-dial delay: The delay between the user signalling a connection request and receiving positive confirmation from the network needs to be short enough that users do not perceive a difference from the post-dial delay they are accustomed to in the circuit switched network, or believe that the network has failed. This is of the order of one second.
- Short post pick-up delay: The delay between a user picking up a ringing phone and the voice path being cut through needs to be short enough so that the "hello" is not clipped. This should be less than a few hundred milliseconds (ideally less than 100 ms).

A key contribution of the Dynamic QoS framework is a recognition of the need for coordination between signalling, which controls access to application specific services, and resource management, which controls access to network-layer resources. This coordination provides a number of critical functions. It ensures that users are authenticated and authorized before receiving access to the enhanced QoS associated with the service. It ensures that network resources are available end-to-end before alerting the destination MTA. Finally, it ensures that the use of

---

[1] ITU-T Rec. G.114 states that a one-way delay of 150 ms is acceptable for most user applications. However, highly interactive voice and data applications may experience degradation even when delays are below 150 ms. Therefore any increase in processing delay (even on connections with transmission times well below 150 ms) should be discouraged unless there are clear service and application benefits.

resources is properly accounted for, consistent with the conventions of traditional voice-grade telephone service (to which some IPCablecom services are similar from a customer perspective) in which charging occurs only after the party receiving a communication picks up.

In order to support the above requirements, the QoS protocols assure that all resources are committed to all transport segments before the signalling protocols cause alerting of the destination. Likewise, during tear down of a session, the QoS protocols include measures to assure that all resources dedicated exclusively to the session are released. Without this coordination between the two directions of data flows, it would be possible for users to thwart the QoS controls and obtain free service. For example, if the paying client terminates the session, but the non-paying does not, a "half channel" remains that can be used to fraudulently transfer data in one direction. The QoS protocols approximate the "all or nothing" transaction semantics for session creation and destruction.

It is desired that the mechanisms used to implement the session be based on existing standards and practices, and also that the results of this work be usable to support alternative call models. These desires have led to the use of the IETF Real Time Protocol (RTP) to carry multimedia data, carried over the IETF User Datagram Protocol (UDP). In-band signalling to set up Quality of Service is carried out using a superset of the IETF Resource reSerVation Protocol (RSVP).

The QoS architecture should provide support for new emerging applications that are dependent on multicast data delivery. Although this is not a strict requirement in the QoS architecture, providing support for multicast will enable the future development of a rich set of multimedia applications. We have not yet examined whether the resource management enhancements introduced here will support multicast seamlessly or not.

For purposes of managing Quality of Service, the bearer channel for a session is managed as three distinct segments: the access network for the originating side of the session, a backbone network, and the access network for the terminating side of the session. J.112 network resources are managed on the basis of J.112 Flows, using the mechanisms defined in ITU-T Rec. J.112. Backbone resources may be managed either per-flow or, more likely, through an aggregated quality of service mechanism. Management of backbone resources is outside the scope of this Recommendation.

Figure 2 graphically shows this model. This Recommendation accommodates a customer environment where a stand-alone MTA may be connected to the CM via a network of links and standard RSVP-capable routers.

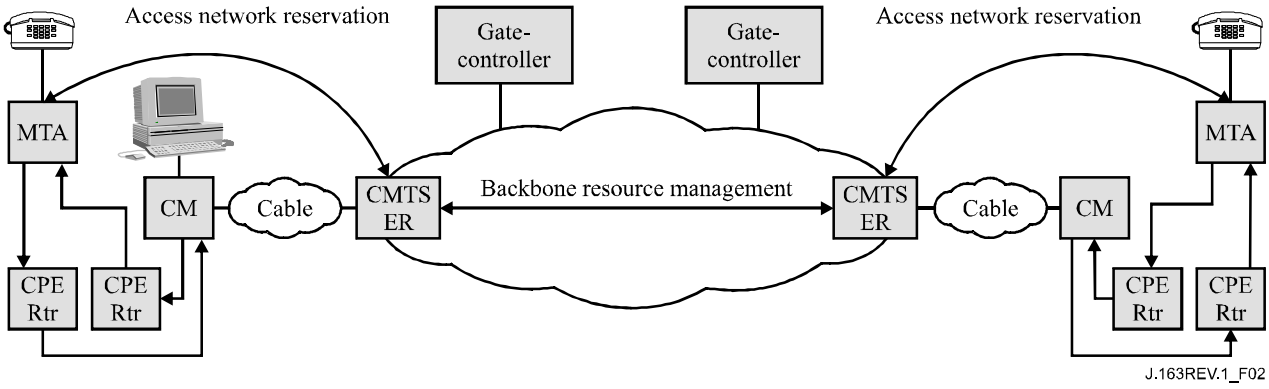


Figure 2/J.163 – Session framework

A QoS-defined construct called a *gate* provides a control point for the connection of access networks to high quality backbone service. A gate is implemented by a CMTS and consists of a packet classifier, a traffic policer, and an interface to an entity that gathers statistics and events (all of these components exist in the J.112 network). A gate can ensure that only those sessions that

have been authorized by the service provider receive high quality service. Gates are managed selectively for a flow. For IPCablecom-based voice communications service, they are opened for individual calls. Opening a gate involves an admission control check that is performed when a resource management request is received from the client for an individual session, and it may involve resource reservation in the network for the session if necessary. The upstream packet filter in the gate allows a flow of packets to receive enhanced QoS for a session from a specific IP source address and port number to a specific IP destination address and port number. The downstream packet filter in the gate allows a flow of packets to receive enhanced QoS for a session from a specific IP source address to a specific IP destination address and port number.

A Gate is a logical entity that resides in a CMTS. A GateID is associated with an individual session and is meaningful at the Gate; the GateID is an identifier that is locally unique at the CMTS, and is assigned by that CMTS. A Gate is unidirectional in nature. If a Gate is "Closed", then data going upstream/downstream on the J.112 access network may either be dropped or provided best-effort service. The choice of dropping packets or serving them on a best-effort basis is a policy choice of the provider.

The gate controller is responsible for the policy decision of when and whether the gate should be opened. A gate is established in advance of a resource management request. This allows the policy function, which is at the gate controller, to be "stateless" in that it does not need to know the state of sessions that are already in progress.

While the gate controls the QoS-guaranteed stream, other flows, such as RTCP or signalling messages, are not policed by the gate. The support of enhanced QoS for signalling messages may play a very important role if the cable system is utilizing high best effort data traffic. In order to meet the signalling performance targets given at the beginning of this clause, it may be crucial to use a dedicated signalling flow with proper QoS constructs. The provisioning specification defines how it is possible to provision the dedicated signalling flow (see Note). It should further be noted that the exact nature of the QoS that should be given to the dedicated signal flow depends on traffic and the CMTS design and is left as a vendor differentiation point.

NOTE – If defined in the configuration file of the MTA the attributes "Call Signalling SCN Up", "Call Signalling SCN Down", and "Call Signalling Network Mask" define the dedicated signalling flow for embedded-MTA.

5.6 Requirements of access network resource management

Providing voice communications service over IP networks with the same level of quality as is available over the PSTN imposes bounds on loss and delay metrics for voice packets and requires active resource management in both the access and backbone networks. The service provider needs to be able to control access to network resources, in order to ensure that adequate capacity is available on an end-to-end basis, even under unusual or overload conditions. The service provider may seek additional revenue for providing a voice communications service with these enhanced quality characteristics (i.e., quality beyond that obtained with a "best-effort" service). The mechanisms provided herein for managed access to enhanced QoS enable the service provider to ensure that access is provided only to authorized and authenticated users on a session-by-session basis and there is no theft of that service.

Clients of the service signal their traffic and performance parameters to the "gate" at the network edge, where the network makes an admission control decision based on both resource availability as well as policy information associated with the gate.

In J.112 networks capacity is limited and it is necessary to do resource management on a per-flow basis. In the backbone there may be several alternatives, ranging from per-flow per-hop admission control to coarse-grained resource provisioning. This Recommendation deals only with access network QoS, and is agnostic about backbone network QoS schemes.

This architecture aims to provide a high degree of generality with the intention of enabling new services and future evolution of network architectures. This goal leads to several requirements for a viable QoS architecture, described in the following subclauses.

5.6.1 Preventing theft of service

The network resources dedicated to the session are protected from misuse, including:

- Authorization and Security: Ensuring that users are authenticated and authorized before receiving access to the enhanced QoS associated with the voice communications service. The CMS/Gate Controller involved in call signalling is trusted to perform these checks and is the only entity which is trusted to create a new gate in a CMTS. The CMS/GC acts as a policy decision point from the perspective of QoS management.
- Resource control: Ensuring that the use of resources is properly accounted for, consistent with the conventions of providers that are part of the PSTN in which charging occurs only after the called party picks up. This includes prevention of utilization of reserved resources for purposes other than the session to which they are assigned. This is achieved through the use of gates and coordination between gates, which bind together address filtering mechanisms with resource reservations.

Since this service may be billed on a per-use basis, there is a significant risk of fraud and service theft. The architecture enables the provider to charge for quality of service. Thus, it prevents theft of service scenarios, several of which are described in Appendix IX.

Theft of service scenarios are addressed in this Recommendation and other Recommendations. They motivate some of the components of the QoS and Call Signalling architectures and protocols.

5.6.2 Two-phase resource commitment

A two-phase protocol for resource commitment is essential to a commercial-grade voice communications service, for two reasons unique to the requirements associated with such a service. First, it ensures that resources are available before signalling the party at the far end that a communication is incoming. Secondly, it ensures that usage recording and billing are not started until the far end picks up, which is also the point at which voice may be cut-through. These properties are provided by conventional telephony signalling protocols; we simply wish to emulate the same semantics here. Also, if bandwidth is allocated before the far end picks up, a theft of service becomes possible. Requiring the endpoints to explicitly send a commitment message ensures that usage recording is based on knowledge of the endpoint and its explicit action.

This framework also supports entities, such as announcement servers and PSTN gateways, that need the voice to be cut through after the first phase of the resource management protocol.

5.6.3 Segmented resource assignment

The Dynamic QoS Architecture partitions resource management into distinct access and backbone segments. Segmented resource assignment is beneficial for two reasons:

- It allows for different bandwidth provisioning and signalling mechanisms for originator's network, far-end network, and backbone network.
- It allows for resource-poor segments to maintain per-flow reservations and carefully manage resource usage. At the same time, when backbone segments have sufficient resources to manage resources more coarsely, it allows the backbone to avoid keeping per-flow state, and thus enhance scalability.

When the backbone does not require explicit per-flow signalling (such as with a Diffserv backbone), it reduces the time taken to set up a session (minimize post-dial delay) and avoids impacting the voice cut-through time (minimize post-pick-up delay).

It potentially reduces the amount of reservation state that is be stored if the remote client is a PSTN gateway.

After the first phase of call signalling, both clients have completed capability negotiation and know what resources are needed end-to-end. Clients send resource management messages using RSVP that may be interpreted hop-by-hop over the local (i.e., user) and access networks (or, optionally for embedded clients, the J.112 MAC-layer Interface). The CMTS maps the resource management messages to the resource management protocol used over the backbone (e.g., IETF diffserv). It also maps the resource management message to the resource management protocol used over the access link (i.e., ITU-T Rec. J.112).

5.6.4 Resource changes during a session

It is possible to change the resources allocated for a session during the life of the session. This facilitates mid-session changes such as switching from a low-rate voice codec to G.711 when modem tones are detected, and the addition of video data to a session that starts as voice only.

5.6.5 Dynamic binding of resources

Dynamic binding of resources (re-reserve) is a requirement to enable efficient use of resources when services such as call waiting are invoked. Abstractly, re-reserving takes bandwidth allocated for a session between a VoIP host and a client and reallocates that same bandwidth to a session with a different client.

It is important to understand the potential danger in de-allocating the session bandwidth, then making a new request for allocation of the new bandwidth. There is a risk of another client using the last remaining bandwidth between the two steps, leaving the original session without an assured quality path. The one-step re-reserve mechanism avoids this, as the bandwidth is not made available to other clients.

5.6.6 Dynamic QoS performance

QoS messaging takes place in real time while callers wait for services to be activated or changed. Thus, the protocol needs to be fast. The number of messages is minimized, especially the number of messages which transits the backbone, and the number of upstream J.112 messages. On the J.112 network, where there is no possibility that forward and reverse paths will be different, this protocol adds several new objects to RSVP, which enables the CMTS to reduce latency by acting as a proxy for the far-end client.

RSVP messages, J.112 management messages, and call signalling messages (collectively referred to as signalling messages) are all transported over the J.112 network on a best effort basis. If the CM is also supporting data services, best effort service may be unable to provide the low latency needed for signalling messages. In this situation, the CM MAY be provisioned with a separate J.112 Flow, with enhanced QoS, to carry signalling traffic. This separate J.112 Flow is provisioned in the same manner as other J.112 media streams, and MAY include classifiers such that its presence is transparent to the MTA.

5.6.7 Session class

Resources may be reserved for different types of services and each service may in turn define different classes of services for its sessions. QoS reservations for sessions designated by the service provider to be of higher priority (e.g., emergency calls) suffer a lower likelihood of blocking than normal sessions. The determination of what session class to assign to a session is performed by the service provider, and is a policy that is exercised by the originating Call Agent/Gate Controller complex at the time the initial session request (e.g., first stage INVITE in the case of DCS) is made.

5.6.8 Intermediate network support

The architecture should not prohibit intermediate networks between the MTA or Multimedia host and the CM (e.g., customer network). Although the intermediate network may not fall under the CABLE OPERATOR's administrative domain or responsibility, allocation of bandwidth in the CABLE OPERATOR's J.112 network is possible when an intermediate network exists. It is also desirable to present a solution that transparently allows for the reservation of resources on the intermediate network.

5.6.9 Backbone QoS support

It is possible that some mechanism for explicitly managing backbone resources will be necessary. The scope of this Recommendation is QoS over the J.112 network, but the architecture provides open, sufficiently general interfaces that are compatible with many of the known backbone QoS mechanisms.

5.6.10 Handling multiple codecs

The NCS signalling used within IPCablecom allows for connections to be established with multiple codecs. In the case where a connection is successfully negotiated with multiple codecs in the list, it is important that the proper resources are allocated to make subsequent codec changes within the list work as expected. Here are the resource components that need to be allocated:

– Authorized Bandwidth: The CMS/GC MUST authorize the Least-Upper-Bound (LUB) of the codec bandwidth that can be used on the connection during Gate allocation.

– Reserved Bandwidth: The MTA MUST reserve the Least-Upper-Bound of the codec bandwidth that can be used during the call (possible codecs are determined from codec negotiation procedure defined in 6.7/J.162).

  NOTE – If the Reserved bandwidth is greater than the committed bandwidth, then the reserved bandwidth needs to be refreshed using J.112 DSCs to the CMTS.

– Committed Bandwidth: The MTA MUST only commit the current codec in use in the upstream direction. This allows the extra unused bandwidth (difference between the Reserved and Committed) to be used for best-effort traffic. In the downstream direction, the MTA MUST commit the Least-Upper-Bound of the codec bandwidth that can be used during the call (possible codecs are determined from codec negotiation procedure defined in 6.7/J.162).

This procedure ensures that a CMS request to switch to any one of the codecs in the negotiated list will be successful. This is especially important in supporting features such as fax/modem that require a switch to G.711 for successful transmission.

If a system provider feels that the above allocation of resources places too much of a constraint on the number of voice channels that can be supported (since we may be over reserving resources in many cases), then the CMS only needs to state a single codec in the LocalConnectionOptions of the connection request. This will ensure that the reserved and committed resources are equal (using the same mechanism as defined in the multiple codec case). Then, if the CMS wants to switch codecs it will need to place the new codec in the LocalConnectionOptions of a subsequent modify connection. However, there are certain risks with this approach. For example, when a modem call is detected and reported to the CMS, it may be possible that the resulting modify connection to use G.711 fails due to insufficient resources on the CMTS. This would not be the case if multiple codecs where defined since the LUB would already be reserved and guaranteed accessible for a subsequent commit.

5.6.11 MTA port-to-port calls

When voice calls are established between different ports (endpoints) on the same MTA, DOCSIS forwarding rules specify that the CM must not forward packets over the DOCSIS network. As a result, the actions taken by the CMS and MTA in this special circumstance are different from a typical MTA-to-MTA call flow. A port-to-port call is defined by both endpoints using the same IP address.

If an MTA receives a connection request without a GateID, it MUST NOT initiate any DSx messaging to the CMTS. If an MTA is instructed to make a port-to-port call, the MTA MUST NOT initiate any DSx messaging to set up a service flow for this connection and MUST NOT send any of the voice packets over the network. In addition, if the MTA has previously created a service flow for a call where the far-end SDP was not available (but a GateID was specified in a CRCX or MDCX), then it MUST subsequently tear down the service flow if a port-to-port call is recognized once the remote SDP is received.

The CMS SHOULD recognize port-to-port calls and SHOULD omit Gate Control to the CMTS and SHOULD omit the GateID in the connection command to the MTA. Similar to the MTA case above, if the CMS has already established a gate for a call where the remote SDP was not available, it SHOULD expect a Gate-Close message from the CMTS once the MTA tears down the service flow upon detection of the port-to-port call. The CMS MUST NOT tear down a call between endpoints with the same IP address on receipt of a GATE-CLOSE message.

5.7 Theory of operation

5.7.1 Basic session set-up

Resource reservation is partitioned into separate Reserve and Commit phases. At the end of the first phase, resources are reserved but are not yet available to the MTA. At the end of the second phase, resources are made available to the MTA and usage recording is started so that the user can be billed for usage.

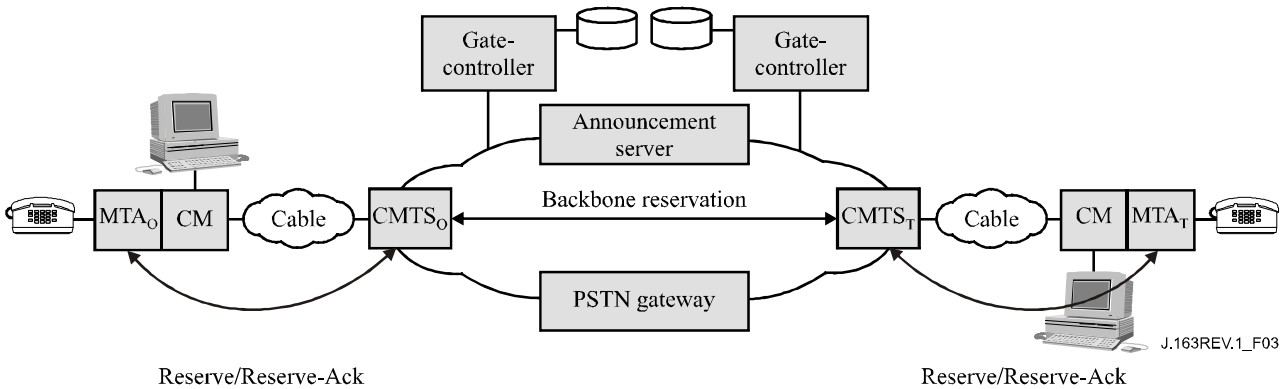


Figure 3/J.163 – Resource management phase 1

Figure 3 shows the first phase of the resource management protocol for a Multimedia application. In this description, subscripts "O" and "T" designate the originating and terminating points of the call. The MTA can be either a stand-alone VoIP host or an embedded MTA; the latter is shown in Figure 3. MTA_O and MTA_T request resource reservation (PATH message in RSVP, or J.112 message in the optional interface for embedded clients) to $CMTS_O$ and $CMTS_T$ respectively. $CMTS_O$ and $CMTS_T$ perform an admission control check for resource availability (initiating signalling for resource reservation in the backbone if necessary) and send a reply to the respective MTAs. In the RSVP framework, the RESV message from the CMTS (where the gate resides) is the acknowledgment to the MTA.

Figure 4 shows the second phase. After determining that resources are available, MTA_O sends a RING message to MTA_T instructing it to start ringing the phone. MTA_T sends a RINGING indication to MTA_O indicating both that resources are available and that the RING message was received. When the called party picks up the phone, MTA_T sends an ANSWERED message to MTA_O and a COMMIT message to $CMTS_T$ When MTA_O receives the ANSWERED message, MTA_O sends a COMMIT message to $CMTS_O$. The COMMIT messages cause resources to be allocated for the call in the J.112 networks. The arrival of the COMMIT messages at $CMTS_T$ and $CMTS_O$ causes them to open their gates, and also starts accounting for resource usage. To prevent some theft of service scenarios, each CMTS informs the respective CMS of the state change by sending a GATE-OPEN message.

The RING, RINGING, and ANSWERED messages shown in this figure and in the above description are logical equivalents to the call signalling messages exchanged by J.162 and SIP IETF RFC 2543.

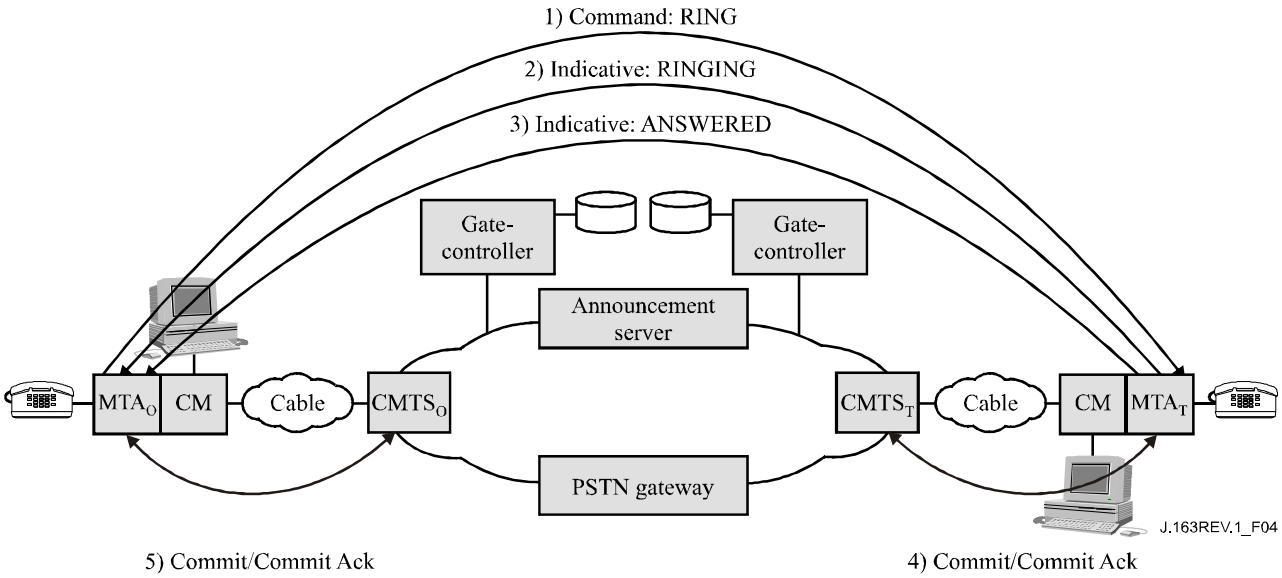


Figure 4/J.163 – Resource management phase 2

5.7.2 Gate coordination

QoS signalling leads to the creation of a gate at each CMTS associated with a client involved in the session. Each gate maintains usage data for the session and controls whether the packets generated by the associated client receive access to enhanced QoS. Gate coordination is needed to prevent fraud and theft of service in situations where a malfunctioning or modified client does not issue the expected signalling messages. It is essential that protocol mechanisms are robust against abuse[2]. A gate coordination protocol ensures that:

- A potential for one-way session establishment without billing is avoided. Because the clients may have adequate intelligence and are not trusted, one can envisage the clients establishing two one-way sessions to provide the users with an adequate interactive voice communication channel. Gate coordination prevents such sessions being established without the provider being able to charge for them.
- The opening and closing of each gate is closely synchronized with the corresponding state changes on the CMS.

---

2 Several theft of service scenarios are described in Appendix IX.

5.7.3 Changing the packet classifiers associated with a gate

Once a pair of gates is set up, clients can communicate over the network with enhanced QoS. Several features needed for a commercial voice communications service involve changing the clients involved in a session, for example when a session is transferred or redirected, or during three-way calling. This requires the packet classifiers associated with a gate to be modified to reflect the address of the new client. In addition, changing the endpoints involved in a session may affect how the session is billed. As a result, gates include addressing information for origination and termination points.

5.7.4 Session resources

The relationship between different categories of resources, authorized, reserved, and committed, is shown in Figure 5. A set of resources is represented by an *n*-dimensional space (shown here as two-dimensional) where *n* is the number of parameters (e.g., bandwidth, burst size, jitter, classifiers) needed to describe the resources. The exact procedures for comparing *n*-dimensional resource vectors are given in ITU-T Rec. J.112.

When a session is first established, DQoS protocols authorize the use of some maximum amount of resources, indicated by the outer oval, specifying the authorized resources. When a client makes a reservation for a session, it reserves a certain amount of resources, which are not greater than those for which it has been authorized. When the session is ready to proceed, the client commits to some amount of resources, which are not more than the reserved resources. In many common cases, the committed and reserved resources will be equal. The committed resources represent resources that are currently in use by the active session, whereas reserved resources represent those that are tied up by the client and have been removed from the pool for admission control purposes, but which are not necessarily being used by the client.

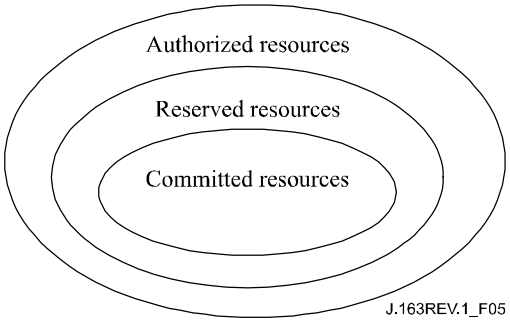


Figure 5/J.163 – Authorized, reserved and committed resources

Authorizations only affect future resource reservation requests. Resources that have been reserved prior to an authorization change are not affected.

Resources that have been reserved but not committed are available to the system for short-term uses only, such as handling of best-effort data. These resources are not available for other reservations (i.e., overbooking is not allowed). The maximum portion of the available resources that can be reserved at once is a policy decision by the CMTS, and outside the scope of DQoS.

Excess resources reserved above those committed are released unless the client explicitly requests they be kept through periodic reservation refresh operations. Maintaining such a condition for long periods of time is discouraged, as it reduces the overall capacity of the system. However, there are situations (e.g., call waiting service, where the call on hold requires resources beyond those needed for the active call) where excess reservations are necessary.

5.7.5 Admission control and session classes

It is envisaged that the Gate at the CMTS may use one or more session classes for resources reserved from an MTA. Session classes define provisionable admission control policies, or their parameters. It is expected that the provider would provision the necessary parameters and/or the alternative admission control policies in the CMTS and in the Gate Controller. For instance, a session class for normal voice communications, and an overlapping session class for emergency calls could be defined to allow the allocation of up to, respectively, 50% and 70% of the total resources to these classes of calls, and leaving the remainder 30-50% of the total bandwidth available to other, possibly lower priority, services. Session classes may furthermore enable pre-emption of already reserved resources, in which case the policy for such pre-emption would be provisionable by the service provider. When the Authorized Envelope is communicated to the Gate at the CMTS by the Gate Controller in the Gate-Set message, the Gate Controller includes adequate information to indicate which session class should apply when the corresponding RESERVE request is processed.

5.7.6 Resource renegotiations

Several of the supported session features require renegotiations of the QoS parameters associated with a session during the lifetime of the session. For example, clients might start communicating using a low-bit-rate audio codec. They can subsequently switch to a higher bit-rate codec or add a video stream, as long as the requested QoS is within the authorized envelope and there is available bandwidth on the network. The use of an authorized QoS envelope that is pre-authorized by the Gate Controller acting as the policy decision point gives clients the flexibility to renegotiate QoS with the network without requiring subsequent Gate Controller involvement. This essentially means that use of resources up to the limits of the envelope is pre-authorized but NOT pre-reserved. Successful allocation of resources within the authorized envelope requires an admission control decision, and is not guaranteed. Subsequent to admission control, the resources are reserved for the flow, although the actual usage of the resources is permitted only after the Commit phase of the Resource Reservation protocol completes. However, no admission control decision is required at the time of commitment of resources. Each change in commitment of resources within the limits of the admission control decision does not require a further reservation. All reservation requests that pass admission control MUST fit within the authorization envelope.

5.7.7 Dynamic binding of resources (Re-reserve)

The Dynamic QoS Architecture recognizes that there may be a need to share resources across multiple sessions, especially when resources are in short supply. In particular, when using the call-waiting feature in telephony-like applications, the client may be involved in two simultaneous sessions, but will be active in only one conversation at a time. It is feasible in this case to share the network-layer resources (in particular, on the access link) between the two conversations. Therefore, this architecture allows a set of network layer resources (such as a bandwidth reservation) to be explicitly identified, and allows one or more gates to be associated with those resources. Signalling primitives allow the resources associated with a gate to be *shared* with another gate at the same CMTS. This improves the efficiency with which resources in the J.112 network are utilized.

When switching back and forth between two sessions in a call-waiting scenario, a client needs to keep enough resources reserved to accommodate either of the sessions, which in general may not need the same amount of resources. Thus, the re-commit operation may change the committed resources. However, the reserved resources do not change in this case, as the client should not have to go through admission control when switching back to the other session.

Whereas the committed resources are always associated with the current active session (and its corresponding IP flow), the reserved resources may be bound to different flows and different gates at different times. A handle, called a resource ID, is used to identify a set of reserved resources for the purpose of binding a flow to those resources.

5.7.8 Support for billing

QoS signalling can be used to support a broad range of billing models, based on only a stream of event records from the CMTS. Since the gate is in the data path, and since it participates in resource management interactions with a client, resource usage accounting is done by the gate. The gate in the CMTS is the appropriate place to do resource accounting, since the CMTS is directly involved in managing resources provided to a client. It is also important to do usage accounting in the CMTS to cope with client failures. If a client that is involved in an active session crashes, the CMTS MUST detect this and stop usage accounting for the session. This can be accomplished using soft state through a resource management refresh message (by having RSVP-PATH messages periodically transmitted for an active session), by monitoring the flow of packets along the data path for continuous-media applications, or by other mechanisms (such as station maintenance) performed by the CMTS. In addition, since the gate retains state for flows that have been authorized by a service-specific Gate Controller, it is used to hold service-specific information related to charging, such as the account number of the subscriber that will pay for the session. The policy function in the Gate Controller thus becomes stateless.

The support required in the CMTS is to generate and transmit an event message to a record keeping server on every change to the QoS, as authorized and specified by a gate. Opaque data provided by the Gate Controller that may be relevant to the record keeping server may also be included in the message. Requirements for handling of event records are contained in other Operations Support specifications.

5.7.9 Backbone resource management

When a CMTS receives a resource reservation message from an MTA, it first verifies that adequate upstream and downstream bandwidth is available over the access channel using locally available scheduling information. If this check is successful, the CMTS can either generate a new backbone resource reservation message, or forward towards the backbone a modified version of the resource reservation message received from the MTA. The CMTS performs any backbone-technology-specific mapping of the resource reservation that is needed. This enables the architecture to accommodate different backbone technologies, at the service provider's choosing. The specific mechanisms for reserving backbone QoS are outside the scope of this Recommendation.

A bidirectional model is used for resource reservation in the J.112 network where the routing is symmetric. A unidirectional model is used for resource reservation in the backbone, which allows routing asymmetries. Thus, when MTA_O makes a reservation with the CMTS, it knows two things: that it has adequate bandwidth in both directions over the J.112 network, and that it has adequate bandwidth over the backbone networks for the MTA_O to MTA_T flow. Thus, MTA_O knows that resources are available end-to-end in both directions once it gets a reply from MTA_T.

5.7.10 Setting the DiffServ code point

This architecture also allows for the use of a Differentiated Services backbone, where there is adequate bandwidth to carry voice conversations, but access to this bandwidth is on a controlled basis. Access to the bandwidth and differentiated treatment is provided to packets with the appropriate encoding of bits in the field of the IP header specified for Differentiated Service. This is called the Diffserv code point (DSCP). The DS field maintains backward compatibility with the present uses of the IP Precedence bits of the IPv4 TOS byte [IETF RFC 2474]. It is desirable to be able to set the Diffserv code point of packets that are about to enter the provider backbone from the CMTS. Since resources consumed by these packets in the backbone may depend heavily on this marking, this architecture provides control of the marking to network entities. This allows the network and service provider to control the use of the enhanced QoS rather than trusting the MTA. The provider can configure policies in the CMTS that determine how to set the DSCP for flows that

pass through the CMTS. Such policies are sent to the CMTS in the gate set-up protocol from the CMS/GC.

For implementation efficiency, we pass the information to the MTA about the appropriate DSCP for it to use on a given session. This is done with the IETF-proposed DCLASS object in RSVP. The CMTS still needs to police received packets to ensure that correct DSCP is being used and that the volume of packets in a given class is within authorized bounds.

5.8 Sample mapping of SDP descriptions into RSVP flowspecs

Session descriptor protocol messages are used to describe multimedia sessions for the purposes of session announcement, session invitation, and other forms of multimedia session initiation per IETF RFC 2327. This clause describes a mechanism for mapping the SDP description into RSVP flow specs.

A typical SDP description contains many fields that contain information regarding the session description (protocol version, session name, session attribute lines, etc.), the time description (time the session is active, etc.), and media description (media name and transport, media title, connection information, media attribute lines, etc.). The two critical components for mapping an SDP description into an RSVP flowspec message are the media name and transport address (m) and the media attribute lines (a).

The media name and transport address (m) are of the form:

m = <media> <port> <transport> <fmt list>

The media attribute line(s) (a) are of the form:

a = <token>:<value>

A typical IP voice communication would be of the form:

m = audio 3456 RTP/AVP 0

a = ptime: 10

On the transport address line (m), the first term defines the media type, which in the case of an IP voice session is audio. The second term defines the UDP port to which the media is sent (port 3456). The third term indicates that this stream is an RTP Audio/Video profile. Finally, the last term is the media payload type as defined in the RTP Audio/Video Profile (IETF RFC 1890). In this case, 0 represents a static payload type of u-law PCM coded single channel audio sampled at 8 kHz. On the media attribute line (a), the first term defines the packet formation time (10 ms).

Payload types other than those defined in IETF RFC 1890 are dynamically bound by using a dynamic payload type from the range 96-127, as defined in IETF RFC 2327, and a media attribute line. For example, a typical SDP message for G.726 would be composed as follows:

m = audio 3456 RTP/AVP 96

a = rtpmap:96 G726-32/8000

The payload type 96 indicates that the payload type is locally defined for the duration of this session, and the following line indicates that payload type 96 is bound to the encoding "G726-32" with a clock rate of 8000 samples/s. For every defined CODEC (whether it is represented in SDP as a static or dynamic payload type), there needs to be a table mapping from either the payload type or ASCII string representation to the bandwidth requirements for that CODEC.

For non-well-known codecs, the bandwidth requirements cannot be determined by the media name and transport address (m) and the media attribute

(a) lines alone. In this situation, the SDP MUST use the bandwidth parameter

(b) line to specify its bandwidth requirements for the unknown codec. The bandwidth parameter line (b) is of the form:

b = <modifier> : <bandwidth-value>

For example:

b = AS:99

This bandwidth parameter along with the media attributes MUST be used to map the SDP into a flowspec, which will be used in the policy authorization decision and subsequent gate allocation.

NOTE – It is a policy decision on the CMS/CMTS whether the requested bandwidth in the SDP is accepted or rejected.

The bandwidth parameter (b) will include the necessary bandwidth overhead for the IP/UDP/RTP headers. In addition, any PHS used in the DOCSIS link will not be reflected in requested bandwidth. In the specific case where multiple codecs are specified in the SDP, the bandwidth parameter should contain the maximum of the desired codec bandwidths.

The mapping of RTP/AVP code to RSVP Flowspec is according to Table 2, of the IPCablecom CODEC specification J.161.

6 MTA to CMTS Quality-of-Service Protocol (pkt-q3)

To meet the requirements described previously, RSVP and the IETF's Integrated Services architecture (IETF RFC 2210) is used as a basis for the signalling mechanism for providing local QoS. RSVP, as currently specified, needs some additional enhancements to meet the requirements of the Dynamic QoS architecture. The combination of RSVP and these extensions is sometimes called RSVP+.

RSVP and the Integrated Services architecture specify QoS parameters in generic terms that are independent of the underlying layer 2 technology. It is necessary to specify a means of mapping those general traffic specifications into specific J.112 Flow specifications. Such mappings exist for other layer 2 protocols (e.g., ATM, IEEE 802.XX LANs); this clause describes mappings for J.112 networks.

The Dynamic QoS Architecture uses a superset of RSVP with the following differences:

- Since resource reservations are independently initiated for each J.112 network (segmented resource allocation model), this Recommendation does not depend on resource management messages propagating end-to-end.
- The resource management exchange between the MTA and CMTS reserves resources in *both* directions over the local area (i.e., customer-operated) and J.112 networks. This allows the CMTS to act as a proxy for the far endpoint, with the benefit of minimizing the number of messages required for resource management in bandwidth-constrained J.112 networks, and reduces the post-dial and post-pick-up delay.
- In the local area (i.e., customer-operated) portion of the network, existing RSVP-capable routers may be present. In this environment, unidirectional reservations are required. To enable these two functions (bidirectional reservations on the J.112 network and unidirectional reservations inside the customer-premises), an enhanced PATH message is issued by the MTA to the Gate.

- Ability to bind a single set of resources to a group of multiple reservations, based on information from the MTA that only one reservation in the group will be active at any given time.
- Support for the two-phase resource activation facility available in ITU-T Rec. J.112, giving the ability to guarantee resources are available before ringing of the far-end phone. The RSVP exchange with the CMTS performs the first stage, the admission control, and the MTA sends a separate message to the CMTS to perform the activation.

The Dynamic Quality of Service operation does not address standard RSVP, which may or may not be supported. Regardless, standard RSVP messages will not trigger the DQoS operations specified in this Recommendation.

6.1 RSVP extensions overview

6.1.1 Segmented operation

As defined in IETF RFC 2205, RSVP is intended to run between a pair of hosts. However, the IPCablecom QoS model requires the signalling to be done in a segmented manner, where one segment is between an MTA and a CMTS. This clause illustrates how RSVP can support a segmented model.

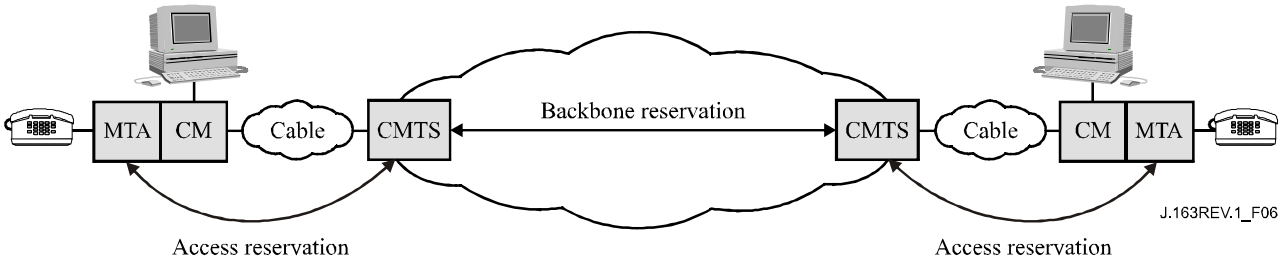


Figure 6/J.163 – Segmented signalling model

In the segmented model, an MTA communicates with the CMTS. In addition to the simple scenario pictured in Figure 6, this Recommendation allows for more complex scenarios, such as when there is a customer network between the client and the CM, which may include a variety of network elements, including RSVP-capable switches or routers. The presence of a customer network means that the solution works even if the client and the CMTS are not immediately adjacent at the IP layer. The customer network may provide multiple paths between the client and the CM, leading to the possibility of asymmetric routes in this network.

The CMTS intercepts RSVP messages sent from the originating MTA to the MTA on the terminating side of the session to implement the Segmented model. This minimizes changes to RSVP, by keeping the destination address of the PATH messages the same as the destination address of the data.

6.1.2 Bidirectional reservations

Traditional RSVP makes unidirectional reservations. PATH messages flow in the same direction as data, and RESV messages flow in the opposite direction. To make a bidirectional reservation, it is necessary to add new RSVP objects to define both directions. The CMTS responds to the request by establishing reservations on both directions of the J.112 link. If there are RSVP-capable routers between the originating MTA and the CM, then the CMTS initiates a PATH message that appears to have come from the remote client.

6.1.3 Header compression, suppression and VAD

If the CMTS and CM are configured to perform header compression or suppression, then the bandwidth that is needed for a J.112 Flow may be reduced. It is necessary for the client to convey to the CMTS the fact that compression or suppression may be applied prior to the installation of a reservation to ensure that appropriate bandwidth is reserved. The general solution to this problem is described in Integrated Services in the Presence of Compressible Flows [IETF RFC 3006].

The MTA adds a parameter (Compression_Hint) described in [IETF RFC 3006] to the Sender-Tspec that identifies the type(s) of header compression or suppression that might be applied to the data. The Compression_Hint parameter contains a Hint field that advertises the type(s) of compression or suppression that is (are) possible, as well as whether the sender is using UDP or IP checksums and/or IP-Ident; if these are not used, these fields may also be compressed or suppressed. If any field in the IP header is not being compressed or suppressed, then the IP checksum MUST NOT be compressed or suppressed.

To signal header suppression to the J.112 network, the CMTS uses the data provided by the Hint field of the Compression_Hint parameter to indicate the scheme of header suppression that will be performed on this J.112 Flow. This information is used to reduce the effective rate and depth of the token bucket supplied by the MTA. If header suppression is not supported on a link, the Compression_Hint parameter is ignored and the full Tspec is used.

When performing header suppression on a J.112 link, it is also necessary to communicate the *contents* of the header that will be suppressed to the CMTS in advance of the first data packet's transmission so that the suppression context can be established at the CM and the CMTS. This information may be delivered by the RSVP message that is used to establish the reservation or through MAC-layer messages sent ahead of the first data packet. Since PATH messages are processed by any intermediate hops between the client and the CMTS, an arriving PATH message will contain the same TTL value as data packets, provided PATH messages and data packets have the same initial TTL when sent by the MTA. The CMTS thus may use the contents of the PATH to learn the values of the fields that will be suppressed. The CMTS uses J.112 MAC-messaging to convey to the CM the fact that suppression should be used for a particular flow, and instructs it to suppress appropriate fields given the presence or absence of UDP checksums.

The CMTS also may instruct the CM to suppress the IP Identification field. This field is used only when fragmentation occurs. Since this field changes with every packet, its value can neither be conveyed using RSVP nor using MAC messages. The question of whether to suppress it or not depends on whether the packet might be fragmented later. There is no need for the MTA to convey any information to the CMTS regarding the suppressibility of this field; the CMTS may decide to suppress it or not based on a local policy.

The same basic approach enables support of Voice Activity Detection (VAD). A CMTS may use different scheduling algorithms for flows that are using VAD, and thus needs to know which flows may be treated with VAD. The compressibility object carried in the Tspec MUST contain a value which indicates that the data flow for which this reservation is being requested may be treated with VAD (i.e., it has not already undergone silence detection at the MTA, and it is voice, not fax or data).

6.1.4 Dynamic binding of resources

The Dynamic QoS model requires the ability to dynamically modify the binding of resources to flows. For example, to provide Call Waiting, it may be desirable to hold enough resources for only one session in place over the J.112 network, and to switch the allocation of those resources from one caller to another. While this capability has been suggested for RSVP in the past, it was not included in RSVP version 1.

In RSVP, the "handle" on a set of reserved resources is the Session-Object. Since the Session contains the destination address of the flow, reallocation of resources to a flow with a different

destination address would require a change in the Session-Object. Changing the source address of the flow could be accomplished using a new Filterspec in the RESV message.

To accommodate this functionality, a Resource-ID object is added to RSVP messages. Routers, which understand this object, will attempt to use the resources associated with that ID. The Resource-ID object is an opaque identifier generated by the node that has control of the resources, i.e., the CMTS in this case.

This is illustrated in Figure 7. When an MTA issues a reservation request for a new flow, it indicates to the CMTS that this session is willing to share resources for this new gate (Gate 2) with a previously created gate (Gate 1) by including the ResourceID in the request. As long as the QoS requested for the new gate can be satisfied with a bandwidth allocation equal to or less than the existing gate, no new bandwidth is reserved in the J.112 network. However, bandwidth may need to be reserved in the backbone network depending on the end-to-end path taken by the new session. Access to the shared reservation occurs in a mutually exclusive manner: an MTA has to issue a commit message to indicate to the CMTS which flow is currently active, and that commit explicitly removes the committed resources for the other. In the call waiting example, the client sends a commit message to the CMTS to identify the currently active flow when the user switches between sessions.

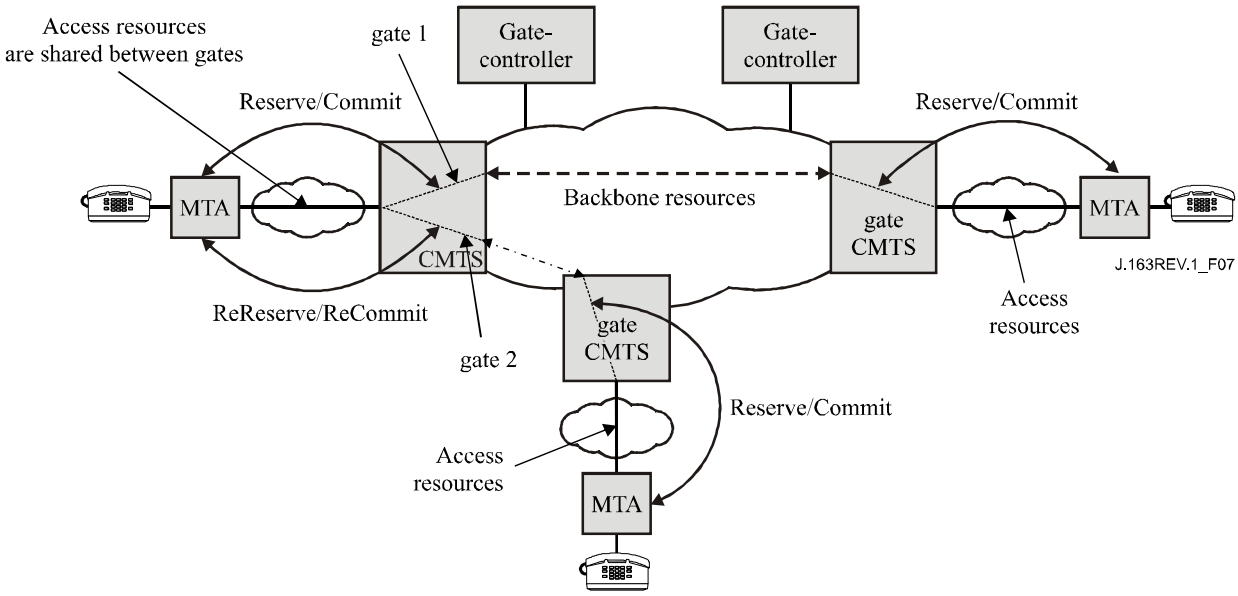


Figure 7/J.163 – Sharing of resource reservations across gates

In the segmented model, the CMTS includes the Resource-ID in the first RESV message that it sends to the MTA. The MTA may include the Resource-ID in subsequent messages that apply to the resources in question. Most importantly, if the MTA wishes to establish a new session, and reuse the resources of an existing session, it includes the Resource-ID associated with the old session in the PATH message it sends to the CMTS. A PATH message that contains the Resource-ID of a currently allocated set of resources adds a new binding between a flow (as identified in the Session and Sender-Template objects) and those resources. It may optionally change the amount of resources allocated by the inclusion of Tspecs and Rspecs which differ from those previously received by the CMTS for this set of resources. This may include the addition of a new set of Tspecs and Rspecs to accommodate multiple codecs as described in 6.2.

RSVP allows reservations to vary in size over time. A reservation that is no greater than the one currently installed (i.e., does not require an increased level of resources in any dimension for either direction of the session) MUST NOT fail admission control. The same rule applies when using the

Resource-ID object. If the amount of resource requested in the new reservation is no greater than previously installed, the reservation MUST NOT fail admission control.

A router that does not understand this new object (e.g., in the customer network) will simply try to install what appears to be a new reservation without reusing previously allocated resources. Since it is unlikely that there is less bandwidth in the home than on the J.112 network, this is unlikely to be a problem. The old reservation will time out if it is not refreshed. In the event that resource scarcity is a problem in the customer network, it would be necessary to upgrade the routers in the home to support this new object. Note that attempting to install reservations on the customer network is worthwhile even if bandwidth is relatively abundant there, as a reservation provides devices in the customer network with the necessary information to isolate specific flows from excessive delay and jitter that they would otherwise experience if simply mixed with best effort traffic (or reserved flows of widely different traffic characteristics) in a common queue.

6.1.5 Two-stage reserve/commit process

A significant aspect of the IPCablecom Dynamic QoS model is that reservation is a two-phase process, with a Commit phase following the Reserve phase. RSVP is used to cover the Reserve phase, so the CMTS does not actually provide the resources until the second stage of the process.

Because the commit phase only involves an MTA and a local gate, it is a unicast message from the MTA to the CMTS. The MTA learns the gate-ID from the call signalling protocol.

6.1.6 Authentication

The provider is able to ensure that parties do not reserve unauthorized network resources. RSVP provides a number of mechanisms to do this, such as RSVP Integrity-Objects and policy data contained in other RSVP messages. The Dynamic QoS specification includes a GateID as policy data, which MUST be included in the RSVP-PATH messages.

6.2 RSVP Flowspecs

The IETF Integrated Services architecture uses general purpose (layer 2 independent) descriptions of the traffic characteristics and resource requirements of a flow. The description of the traffic is known as a Tspec, the resource requirements are contained in an Rspec, and the combination of these is known as a Flowspec. In order to reserve resources on a specific layer 2 medium such as a J.112 network, it is necessary to define a mapping from the layer 2 independent Flowspec to specific layer 2 parameters. Mappings for a variety of other technologies (ATM, 802.3 LANs, etc.) have already been defined.

Other specifications (e.g., the IPCablecom CODEC specification J.167) contain the mapping requirements of higher-layer service descriptions (e.g., SDP as used in VoIP applications) into Flowspecs. This clause specifies how the CMTS and MTA MUST map Flowspecs to DOCSIS layer 2 parameters.

Integrated Services currently defines two types of service, controlled load and guaranteed, the latter being the more suitable for latency sensitive applications. When making a reservation for guaranteed service, the Flowspec contains:

Tspec

Bucket depth (b) – bytes
bucket rate (r) – bytes/second
peak rate (p) – bytes/second
min policed unit (m) – bytes
maximum datagram size (M) – bytes

Rspec

reserved rate (R) – bytes/second
slack term (S) – microseconds

The Tspec terms are mostly self-explanatory. (r,b) specifies a token bucket that the traffic conforms to, p is the peak rate at which the source will send, and M is the maximum packet size (including IP and higher layer headers) that will be generated by the source. The minimum policed unit, m, is usually the smallest packet size that the source will generate; if the source sends a smaller packet, it will count as a packet of size m for the purposes of policing.

To understand the Rspec, it is helpful to understand how delay is calculated in an Integrated Services environment. The maximum end-to-end delay experienced by a packet receiving Guaranteed service is:

$$\text{Delay} = b / R + C_{tot} / R + D_{tot}$$

where b and R are as defined above, and C_{tot} and D_{tot} are accumulated "error terms" provided by the network elements along the path, which describe their deviation from "ideal" behaviour.

The rate R provided in the Rspec is the amount of bandwidth allocated to the flow. It MUST be greater than or equal to r from the Tspec for the above delay bound to hold. Thus, a flow's delay bound is completely determined by the choice of R; the reason to use a value of R greater than r would be to reduce the delay experienced by the flow.

Since it is not permissible to set R < r, a node making a reservation may perform the above calculation and determine that the delay bound is tighter than needed. In such a case, the node may set R = r and set S to a non-zero value. The value of S would be chosen such that:

$$\text{Desired delay bound} = S + b / R + C_{tot} / R + D_{tot}$$

Guaranteed Service does not attempt to bound jitter any more than is implied by the delay bound. In general, minimum delay that a packet might experience is the speed of light delay, and the maximum is the delay bound given above; the maximum jitter is the difference between these two. Thus jitter may be controlled by suitable choice of R and S.

6.2.1 Complex SDP descriptions with multiple codecs

There are various situations in which a reservation needs to cover a range of possible flowspecs. For example, for some applications it is desirable to create a reservation, which can handle a switch from one codec to another mid-session without having to pass through admission control at each switch-over time.

The Sender Tspec MUST contain the Least Upper Bound (LUB) of the necessary flow parameters for the component flow.

The Least Upper Bound (LUB) of two flows A and B, LUB(A, B), is the "smallest" envelope that can carry both of the flows A, B non-simultaneously. LUB(A, B) is calculated on a parameter-by-parameter basis as follows:

Define the Tspec values for a flow α as in 6.2. Also define the period Pα as Mα/rα. Then LUB(A, B) is given by:

LUB(A, B) ≡ { bLUB(A, B) ≡ MAX(bA, bB),
r LUB(A, B) ≡ (M LUB(A, B)/P LUB(A, B)),
p LUB(A, B) ≡ MAX(pA, pB, r LUB(A, B)),
m LUB(A, B) ≡ MAX(mA, mB),
M LUB(A, B) ≡ MAX(MA, MB)
}

where:

p LUB(A, B) ≡ GCF(PA, PB);

the function MAX(x, y) means "take the higher of the pair (x, y)";

the function MAX(x, y, z) ≡ MAX(MAX(x, y), z);

the function GCF(x, y) means "take the Greatest Common Factor of the pair (x, y)".

The LUB of n flows (n ≠ 2), LUB(n1, n2, ...), is defined recursively as:

$$LUB(n1, n2, ..., N) \equiv LUB(n1, LUB(n2, ..., N))$$

In addition, the slack term in the corresponding Rspec must allow any component flow to use the resources. In order to ensure that this criterion is met, the Rspec for the flow is set to the minimum value of the Rspec values in the component flows. That is:

$$SLUB(A, B) \equiv MIN(SA, SB)$$

where the function MIN(x, y) means "take the lower of the pair (x, y)".

The following example shows how Tspec parameters are determined using LUB algorithm specified above:

1) As the result of codec negotiation, the following codecs are selected for a call:

   G711(20ms) and G728(10ms)

2) The LUB bucket depth for the selected codecs is:

   G711(20ms) = (8000/50) + 40 = 200 bytes

   G728(10ms) = (2000/100) + 40 = 60 bytes

   b[LUB] = m[LUB] = M[LUB] = MAX(200, 60) = 200 bytes

3) The LUB bucket rate for selected codecs is:

   P [LUB] = GCF(10ms, 20ms) = 10ms = 0.01 second

   r[LUB] = M × 1/P = 200 × 1/0.01 = 20,000 bytes per second

   r[G711(20ms)] = 200 × 1/0.02 = 10,000 bytes per second

   r[G728(10ms)] = 60 × 1/0.01 = 6,000 bytes per second

   p[LUB] = MAX(10000, 6000, 20000) = 20,000 bytes per second

6.2.2 Mapping PacketCable codecs into DQoS RSVP requests

The following DQoS profile of the DOCSIS 1.1 QoS mechanisms must be used in support of voice services. Due to the fact that the IPCablecom defined codecs (per ITU-T Rec. J.161) are transported using a constant bit-rate data stream, the following rules MUST be applied in constituting DQoS messages:

The parameters *RSVP Bucket Depth* (b), *RSVP Maximum Datagram Size* (M), and *RSVP Minimum Policed Unit* (m) MUST be equal to each other.

The parameters *RSVP Bucket Rate* (r), *RSVP Peak Rate* (p) and *RSVP Reserved Rate* (R) MUST be equal to each other.

The *RSVP Slack Term* MUST be set to a value supplied by the CMS, if the value is not supplied by the CMS then the default values of 800 microseconds for upstream and the value zero for downstream MUST be used.

The *RSVP Protocol* MUST be set to UDP.

The *RSVP Destination Address* MUST be set to IP address to which the data stream packets will be sent, if known for the direction that the parameter is being used. If the parameter is not known, then the value zero MUST be used in this field.

The *RSVP Destination Port* MUST be set to UDP port to which the data stream packets will be sent, if known for the direction that the parameter is being used. If the parameter is not known, then the value zero MUST be used in this field.

The *RSVP Source Address* MUST be set to IP address from which the data stream packets will be sent, if known for the direction that the parameter is being used. If the parameter is not known, then the value zero MUST be used in this field.

The *RSVP Source Port* MUST be set to UDP port from which the data stream packets will be sent, if known for the direction that the parameter is being used. If the parameter is not known, then the value zero MUST be used in this field.

If an entity receives a DQoS message that does not conform to the requirements outlined in this Recommendation, then the entity MUST permanently reject the request.

6.2.3 Mapping RSVP Flowspecs into J.112 QoS parameters

The CMTS, on receiving a reservation request, decides:

- what type of J.112 service to use (e.g., unsolicited grant, real-time polled, etc.);
- what QoS parameters to associate with the corresponding Service Flow.

The RSVP objects for both directions must be set as follows:

Sender Tspec and *Reverse Sender Tspec*:

Bucket Depth (b), bytes = VoIP datagram size, including IP/UDP/RTP header overhead.

Bucket Rate (r), bytes/second = actual data rate, including IP/UDP/RTP header overhead.

Maximum Datagram Size (M), bytes = Bucket Depth (b).

Minimum Policed Unit (m), bytes = Bucket Depth (b).

Peak Rate (p), bytes/second = Bucket Rate (r).

Sender Rspec and *Reverse Sender Rspec*:

Reserved Rate (R), bytes/second = Bucket Rate (r).

Slack Term (s), microseconds = tolerated grant jitter for upstream flows, tolerated latency for downstream flows[3]. This value is in the range of $0 \leq s \leq 2 \times$ packetization interval with a default value of 800 microseconds for upstream, and in the range $50 \leq s \leq 2 \times$ packetization interval or a (default) value of zero for downstream direction if these values are not specified by the CMS to the MTA. In the downstream direction, zero indicates no restrictions on latency.

Session and *Reverse Session*:

Protocol = UDP.

Destination Address = IP address to which the data stream packets will be sent.

Destination Port = UDP port to which the data stream packets will be sent.

Sender Template and *Reverse Sender Template*:

Source Address = IP address from which the data stream packets will be sent.

Source Port = UDP port from which the data stream packets will be sent.

The same values MUST be used when committing the resources.

---

[3] The actual value specified depends upon the end-to-end latency budget for a particular call, and the CMS may specify a value based on routing information (e.g., propagation delay to destination).

6.2.3.1 Upstream quality of service encodings

The DOCSIS upstream objects must be set as stated below. All the other service flow quality-of-service TLV encodings MUST NOT be defined, thereby allowing the default values to be used. If the MTA provides one of these TLVs, then the CMTS MUST reject the request with a "reject permanent/reject admin" error code.

The *DOCSIS Active Timeout* timer value is used to detect inactivity and initiate resource recovery for committed service flows. MTA/CMTS synchronization may be coordinated by the CMTS by providing an appropriate value in the DSA/DSC REQ/RSP message. This field MUST NOT be populated by the MTA.

The *DOCSIS Admitted Timeout* timer value is used to detect inactivity and initiate resource recovery for reserved service flows. MTA/CMTS synchronization may be coordinated by the CMTS by providing an appropriate value in the DSA/DSC REQ/RSP message. This field MUST NOT be populated by the MTA.

The *DOCSIS Assumed Minimum* Reserved Rate Packet Size parameter MUST NOT be set for upstream flows.

The *DOCSIS Grants per Interval* parameter MUST be set to 1.

The *DOCSIS Nominal Grant Interval* parameter MUST be set to the codec packetization interval.

DOCSIS Nominal Grant Interval = 10000 or 20000 or 30000

The *DOCSIS Tolerated Grant Jitter* parameter MUST be set to a CMS-specified value which is based on routing cost information. Allowed range for this parameter is between 0 and 2 × packetization interval. If the value is not specified by the CMS, a default value of 800 microseconds MUST be used.

The *DOCSIS Nominal Polling Interval* parameter MUST NOT be specified for UGS service flows, and SHOULD be set to a value that is integer multiple of the codec packetization interval for UGS/AD service flows.

The *DOCSIS Tolerated Polling Jitter* parameter MUST NOT be specified for UGS service flows, and SHOULD be set to a value that is integer multiple of the codec packetization interval for UGS/AD service flows.

The *DOCSIS Request/Transmission Policy* parameter is a bitmask and bits 0-6 and 8 MUST be set for UGS and UGS/AD service flows.

The *DOCSIS TOS Overwrite* parameter MUST NOT be used. Even though this parameter is defined by DOCSIS, the use of the field is prohibited by PacketCable.

The *DOCSIS Unsolicited Grant Size* parameter MUST be calculated from the DOCSIS MAC header FC to end of CRC. The value includes Ethernet header overhead of 18 bytes (6 bytes for source address, 6 bytes for destination address, 2 bytes for length, and 4 bytes for CRC). The value also incorporates DOCSIS MAC layer overhead, including the DOCSIS base header (6 bytes), the UGS extended header (3 bytes), and the BPI+ extended header (5 bytes). If payload header suppression (PHS) is active then the number of suppressed bytes MUST NOT be included. Note that the PHS extended header (2 bytes) MUST NOT be included for UGS or UGS/AD service flows, since the appropriate information is embedded in the UGS extended header.

DOCSIS Unsolicited Grant Size^8,9 = M + 32-PHS[4][5]

---

4 This example assumes that BPI+ is being used as mandated by the PacketCable Security specification.

5 The PHS used in this example is defined in the DOCSIS RFI specification, clause B.C.2.2.10.4 of Annex B to J.112.

The *DOCSIS Upstream Scheduling Type parameter* MUST be set to either UGS or UGS/AD, depending on whether silence suppression is supported on the call.

If the MTA is making a reservation or commit for a codec that does not perform Voice Activity Detection then the MTA MUST use the UGS as scheduling type, otherwise it MUST use UGS/AD.

If the MTA is making a reservation for a Service Flow for multiple codecs of which one of them will perform Voice Activity Detection, then the MTA MUST request the UGS/AD for reservation and commit for only the active codec's properties as described in the above paragraph.

6.2.3.2 Upstream packet classification encodings

DOCSIS upstream packet classification requests

The DOCSIS upstream objects must be set as stated below. All the other classification TLV encodings MUST NOT be defined, thereby allowing the default values to be used. If the MTA provides one of TLVs that are to be omitted, then the CMTS MUST reject the request with a "reject permanent/reject admin" error code.

If defined by the CMTS, the *DOCSIS Classifier Identifier* parameter MUST be used. Otherwise, the *DOCSIS Classifier Reference* parameter MUST be set to a unique value per Dynamic Service Message.

The *DOCSIS Service Flow* Reference parameter MUST be set to an E-MTA unique value for existing calls for DSA_REQ messages, and MUST be omitted in all other messages. Instead, the CMTS-issued DOCSIS Service Flow Identifier parameter MUST be used.

The *DOCSIS Rule Priority* parameter MUST be set to 128.

The *DOCSIS Classification Activation* State parameter MUST be set to active (1) when the call utilizing the service flow is committed, and for all the other cases it MUST be set to inactive (0).

The *DOCSIS Dynamic Service Change Action* MAY use the DSC Add Classifier (0), DSC Replace Classifier (1) and DSC Delete Classifier (2) operations per the DOCSIS RFI specification.

The *DOCSIS IP TOS* and mask fields MAY be omitted, since PacketCable does not incorporate TOS parameters as part of its classifier. Alternatively, if this parameter is included it MUST correspond with the TOS value specified by the CMS or a provisioned value for voice service flows.

The *DOCSIS IP Protocol* parameter MUST be set to UDP (17).

The *DOCSIS IP Source Address* parameter MUST be set to the same address as that in the Sender Template, so long as a non-zero value is provided. If the address specified in the Sender Template object is zero, this parameter MUST be omitted.

The *DOCSIS IP Source Mask* parameter MUST be omitted.

The *DOCSIS IP Source Port Start* and *DOCSIS IP Source Port End* parameters MUST be set to the same transport port value as the Sender Template.

The *DOCSIS IP Destination Address* parameter MUST be set to the same address as that in the Session Object, so long as a non-zero value is provided. If the address specified in the Session Object is zero, this parameter MUST be omitted.

The *DOCSIS IP Destination Mask* parameter MUST be omitted.

The *DOCSIS IP Destination Port Start* and *DOCSIS IP Destination Port End* parameters MUST be set to the same transport port as the Session Object, so long as a non-zero value is provided. If the Destination IP Port is specified as a value of zero in the Session Object, then the DOCSIS IP Destination Port Start and End TLVs MUST be omitted.

The *DOCSIS Ethernet LLC Packet Classification Encodings* parameters MUST be omitted.

The *DOCSIS 802.1P/Q Packet Classification Encodings* parameters MUST be omitted.

CMTS behaviour for DOCSIS upstream packet classification requests

Upon Reception of the Classifier Addition request (e.g., via DOCSIS DSx messaging) the CMTS MUST compare the Gate settings referenced by the GateID to the TLVs. If the TLVs do not match, the CMTS MUST return the DOCSIS Classifier Error Encoding with following information:

- The *Error Code* parameter MUST contain a "reject-authorization-failure" value.
- The *Errored Parameter* parameter MUST reference the first TLV that failed authorization. Since different implementations MAY authenticate the TLVs in different order, the TLV returned in this field MAY be different under identical conditions.
- The *Error Message* parameter MAY be populated.

6.2.3.3 Payload header suppression encodings

DOCSIS payload header suppression requests

Payload Header Suppression is optional; however, if used, the requirements below must be followed. These rules apply to PHS on both upstream and downstream flows.

The *DOCSIS Payload Header Suppression Field* parameter references the bytes of the headers which MUST be suppressed by the sending entity, and MUST be restored by the receiving entity.

The *DOCSIS Payload Header Suppression Size* parameter MUST be equal to the total number of bytes in the Payload Header Suppression Field (PHSF).

The *DOCSIS Payload Header Suppression Mask* parameter MUST indicate the bytes to be suppressed.

The *DOCSIS Payload Header Suppression Verification* parameter SHOULD be set to 0 (verify).

The *DOCSIS Classifier Identifier* parameter MUST be used if defined by the CMTS. Otherwise, the *DOCSIS Classifier Reference* parameter that was used in the classifier definition MUST be used.

The *DOCSIS Classifier Reference* parameter MUST be used if DOCSIS Classifier Identifier is not defined by the CMTS. Otherwise, the DOCSIS Classifier Identifier parameter that was used in the classifier definition MUST be used.

The *DOCSIS Service Flow* Identifier parameter MUST be used if defined by the CMTS. Otherwise, the DOCSIS Service Flow Reference parameter that was used in the classifier definition MUST be used.

The *DOCSIS Dynamic Service Change* Action MAY use the Add PHS Rule (0), Set PHS Rule (1) Delete PHS Rule (2), and Delete All PHS Rules operations per the DOCSIS RFI specification.

CMTS behaviour for DOCSIS payload header suppression requests

The PHS error handling described here provides a fairly sophisticated feedback mechanism between the CMTS which rejects an initial PHS request and the requesting MTA with the intent that the information provided in the error response may be used to facilitate a successful alternative approach (i.e., the successful admission of the UGS flow without suppression or with a simpler PHS rule).

Upon reception of DSx request with DOCSIS Payload Header Suppression, if a CMTS decides that it cannot support the requested suppression (perhaps due to a lack of local processing or memory resources) but can support the Unsolicited Grant Service without suppression, it MUST return the confirmation code "reject-header-suppression" in the DOCSIS Payload Header Suppression Error Encodings along with the DOCSIS Errored Parameter as described below. The DOCSIS Error Message MUST NOT be used.

If the CMTS cannot support a requested complex DOCSIS Payload Header Suppression, but can support a simpler one then the CMTS MUST provide the DOCSIS Payload Header Suppression Mask in the DOCSIS Errored Parameter field.

DOCSIS Errored Parameter = DOCSIS Payload Header Suppression Mask

If the CMTS cannot support a requested complex size for the DOCSIS Payload Header Suppression but can support a smaller DOCSIS Payload Header Suppression Size, then the CMTS MUST provide the DOCSIS Payload Header Suppression Size in DOCSIS Errored Parameter field.

DOCSIS Errored Parameter = DOCSIS Payload Header Suppression Size

E-MTA behaviour for DOCSIS payload header suppression requests

Upon reception of a "reject-header-suppression" confirmation code in which the DOCSIS Errored Parameter includes the DOCSIS Payload Header Suppression Mask, the E-MTA MAY re-request the bandwidth without DOCSIS Payload Header Suppression or MAY redefine the DOCSIS Payload Header Suppression Mask such that the mask would contain a simpler suppression rule (e.g., indicating a contiguous block of suppressed bytes).

Upon reception of a "reject-header-suppression" confirmation code in which the DOCSIS Errored Parameter includes the DOCSIS Payload Header Suppression Size, the E-MTA MAY re-request the bandwidth without DOCSIS Payload Header Suppression.

E-MTA use of the DOCSIS UGS extended header

The DOCSIS Payload Header Suppression Index parameter MUST contain the pre-established PHS index value or zero when there are no Payload Header Suppression defined for the Service Flow.

The DOCSIS Queue Indicator parameter MUST be set by the E-MTA whenever more than one packet has been queued for transmission. Otherwise, this value SHOULD be cleared to zero.

The DOCSIS Active Grants parameter MUST be set to one whenever the E-MTA is not in the Silence Suppression, and MUST be set to zero whenever the E-MTA is in Silence Suppression for the codec that is being used for the data stream associated with this Service Flow.

6.2.3.4 Downstream Quality of Service encodings

The DOCSIS downstream service flow quality-of-service TLV encodings MUST be set as stated below. All other TLVs MUST NOT be defined, thereby allowing the default values to be used. If the MTA uses one of these TLVs, then the CMTS MUST reject the request with a "reject permanent/reject admin" error code.

The downstream DOCSIS parameters are calculated from the DOCSIS MAC header byte following the HCS to end of CRC. The MAC layer (i.e., Ethernet) overhead is 18 bytes (6 bytes for source address, 6 bytes for destination address, 2 bytes for length, and 4 bytes for CRC).

Based on this overhead, *the DOCSIS Assumed Minimum Reserved Rate Packet Size* parameter MUST be calculated as:

DOCSIS Assumed Minimum Reserved Rate Packet Size = m + 18 – PHS

The *DOCSIS Maximum Sustained Traffic Rate*[6] parameter is given in bits per second, including Ethernet (not DOCSIS) MAC layer overhead. The conversion from IP-specific parameters involves first determining the packetization rate by dividing the Peak Rate by the Minimum Policed Unit. This value is then multiplied by the packet size, amended to include MAC layer overhead, and the entire product is scaled from bytes to bit. The DOCSIS maximum sustained traffic rate MUST be calculated as:

$$\text{DOCSIS Maximum Sustained Traffic Rate} = p / m \times (m + 18 - PHS) \times 8$$

The DOCSIS Minimum Reserved Traffic Rate[6] parameter is calculated in a manner similar to the DOCSIS Maximum Sustained Traffic Rate, except that instead of using the Peak Rate Parameter (p), the Reserved Rate (R) is used.

$$\text{DOCSIS Minimum Reserved Traffic Rate} = R / m \times (m + 18 - PHS) \times 8$$

The *DOCSIS Maximum Traffic Burst* parameter MUST be set to the greater of:

1) an integer multiple of Assumed Minimum Reserved Rate Packet Size; or
2) the DOCSIS specified minimum value of 1522.

$$\text{DOCSIS Maximum Traffic Burst} = \max((M + 18 - PHS) \times 3, 1522)$$

The *DOCSIS Traffic Priority* parameter MUST be set to five.

The *DOCSIS Downstream Latency* parameter MUST NOT be used.

The *DOCSIS Active Timeout* timer value is used to detect inactivity and initiate resource recovery for committed service flows. Because both upstream and downstream service flows and Gates are managed under a single GateID and are deleted in pairs, in the PacketCable model it is not necessary to monitor both upstream and downstream flows for activity. For this reason, only upstream service flows are monitored through the use of the DOCSIS Active Timeout value. This field MUST NOT be populated by the MTA or CMTS for downstream service flows.

The *DOCSIS Admitted Timeout* timer value is used to detect inactivity and initiate resource recovery for reserved service flows. However, by the same logic as described above for the DOCSIS Active Timeout parameter, monitoring of downstream service flows through the use of the DOCSIS Admitted Timeout parameter is not defined in the IPCablecom model. This field MUST NOT be populated by the MTA or CMTS for downstream service flows.

6.2.3.5 Downstream Packet Classification Encodings

DOCSIS downstream packet classification requests

The DOCSIS downstream classification objects MUST be set as stated below. All the other classification TLV encodings MUST NOT be defined, thereby allowing the default values to be used. If the MTA includes one of TLVs that are to be omitted, then the CMTS MUST reject the request with a "reject permanent/reject admin" error code.

If defined by the CMTS, the *DOCSIS Classifier Identifier* parameter MUST be used. Otherwise, the *DOCSIS Classifier Reference* parameter MUST be set to a unique value per Dynamic Service Message.

The *DOCSIS Service Flow Reference* parameter MUST be set to an E-MTA unique value for the existing calls for DSA_REQ messages, and MUST be omitted in all other messages. Instead, the CMTS-issued *DOCSIS Service Flow Identifier* parameter MUST be used.

The *DOCSIS Rule Priority* parameter MUST be set to 128.

---

6 It should be noted that if a value has a fractional value, then it has to be rounded up.

The *DOCSIS Classification Activation State* parameter MUST be set to active (1) when the call utilizing the service flow is committed, and for all the other cases it MUST be set to inactive (0).

The *DOCSIS Dynamic Service Change Action* MAY use the DSC Add Classifier (0), DSC Replace Classifier (1) and DSC Delete Classifier (2) operations per the DOCSIS RFI specification.

The *DOCSIS IP TOS* and mask fields MUST NOT be used.

The *DOCSIS IP Protocol* parameter MUST be set to UDP (17).

The *DOCSIS IP Source Address* parameter MUST be set to the same address as that in the Reverse Sender Template, so long as a non-zero value is provided. If the address specified in the Reverse Sender Template object is zero, this parameter MUST be omitted.

The *DOCSIS IP Source Mask* parameter MUST be omitted.

The *DOCSIS IP Source Port Start* and *DOCSIS IP Source Port End* parameters MUST be set to the same transport port value as indicated in the Reverse Sender Template, so long as a non-zero value is provided. If the Source IP Port is specified as a value of zero in the Reverse Sender Template, then the DOCSIS IP Source Port Start and End TLVs MUST be omitted.

The *DOCSIS IP Destination Address* parameter MUST be set to the same address as indicated in the Reverse Session object.

The *DOCSIS IP Destination Mask* parameter MUST be omitted.

The *DOCSIS IP Destination Port Start* and *DOCSIS IP Destination Port End* parameters MUST be set to the same port as indicated in the Reverse Session object.

The *DOCSIS Ethernet LLC Packet Classification Encodings* MUST be omitted.

The *DOCSIS 802.1P/Q Packet Classification Encodings* MUST be omitted.

CMTS behaviour for DOCSIS downstream packet classification requests

Upon Reception of the Classifier Addition request (e.g., via DOCSIS DSx messaging) the CMTS MUST compare Gate settings referenced by the GateID to the TLVs of the request. If the TLVs do not match, the CMTS MUST return a DOCSIS Classifier Error Encoding with following information:

- The *Error Code* parameter MUST contain "reject-authorization-failure".
- The *Errored Parameter* parameter MUST point to the first TLV that failed authorization. Since different implementations may authenticate TLVs in different order, the TLV returned in this field may be different under identical conditions.
- The *Error Message* parameter MAY be populated.

6.2.3.6 Example of mapping

Consider the following example. A voice codec produces a CBR output data stream of 64 kbit/s which is packetized at 10 ms intervals, thus producing an 80-byte payload each 10 ms. The payload is encapsulated using RTP/UDP/IP, an extra 40 bytes, yielding a 120-byte packet each 10 ms. The Tspec in this case is:

bucket depth (b) = 120 bytes

bucket rate (r) = 12 000 bytes/second

peak rate (p) = 12 000 bytes/second

min policed unit (m) = 120 bytes

maximum datagram size (M) = 120 bytes

Suppose a client requests a reservation using this Tspec and an Rspec with R = r. A CMTS receiving this request will establish a Service Flow that uses Unsolicited Grant Service because p = r and M = b, indicating a CBR flow. It may use a grant size of M bytes at an interval of M/R = 10 ms.

For the calculation of jitter, the MTA does not know how much the CMTS deviates from ideal in its scheduling behaviour. The client should assume that the CMTS is ideal, which means that the delay it will experience with the above Tspec and its reserved rate R = r is simply:

b/r + propagation delays

Ignoring the propagation delay, this results in a delay of 10 ms. Suppose that the client is willing to tolerate a 15 ms delay for this session (on the client-CMTS path only), it would then set its slack term (S) to 15 – 10 = 5 ms. On receiving the reservation, the CMTS interprets this as an indication that a 5 ms grant jitter is acceptable to the client.

Suppose that the client is willing to tolerate a 25 ms delay, and sets its slack term to 25 – 10 = 15 ms. The CMTS may use this information to determine that it can use a longer grant interval, e.g., 20 ms, since this potentially increases delay up to 20 ms for a packet that arrives at the CM right after a grant. There is still 5 ms of slack left, which the CMTS may use to set the grant jitter.

Note that this approach leaves considerable flexibility in the CMTS to meet the requirements of the client with regard to delay in whatever way best matches the capabilities of the CMTS.

6.2.3.7 Payload header suppression and VAD

If the CMTS and CM perform header suppression, then the bandwidth that is needed on a Service Flow can be reduced. The client MUST convey to the CMTS the fact that suppression may be applied prior to the installation of a reservation to ensure that appropriate bandwidth is reserved. The general solution to this problem is described in IETF RFC 3006. The sender (client) adds a parameter (Compression_Hint), described in IETF RFC 3006, to the Sender-Tspec that identifies the type of compression or header suppression that might be applied to the data. The Compression_Hint parameter contains a Hint field that advertises the type(s) of compression that is possible.

An MTA that desires the CM to perform header suppression MUST include the Compression_Hint parameter, IETF RFC 3006, in the Tspec. The Compression factor field, a percentage in the range 1 to 100 inclusive, MUST be set to an amount that yields the bandwidth savings when PHS (42 bytes) is used. The value for Compression factor varies relative to the traffic profile of the CODEC. The Hint MUST be set to one of the following values depending on the type(s) of compression/suppression the MTA desires:

0x40090001 Do not suppress UDP checksum AND Do not suppress IP-Ident field nor IP-Checksum field.

0x40090002 Do not suppress UDP checksum AND suppress IP-Ident field and IP-Checksum field.

0x40090003 Suppress UDP checksum AND Do not suppress IP-Ident field nor IP-Checksum field.

0x40090004 Suppress UDP checksum AND suppress IP-Ident field and IP-Checksum field.

Note that suppression of the IP-Ident field will create problems if the packet is subsequently fragmented within the IP network. For packets less than 576 bytes in length (Internet default value of MAX-MTU), it is reasonable to assume no fragmentation will occur. The MTA SHOULD NOT request the IP-Ident field be suppressed if it will be sending packets longer than 576 bytes.

A CMTS connected to a CM that is capable of performing header suppression uses the Compression_Hint parameter [IETF RFC 3006] to reduce the effective rate and depth of the token

bucket supplied by the sender. If header suppression is not supported on a link, the Compression_Hint parameter is ignored and the full Tspec is used.

When performing header suppression on a J.112 link, it is also necessary to communicate the *contents* of the header that will be suppressed to the CMTS in advance of the first data packet's transmission so that the suppression context can be established at the CM and the CMTS. All this information is present in the RSVP message that is used to establish the reservation, including source and destination IP addresses and ports. Since PATH messages are processed by any intermediate hops between the client and the CMTS, an arriving PATH message will contain the same TTL value as data packets, provided PATH messages and data packets have the same initial TTL when sent by the client. The CMTS MUST use the contents of the PATH to learn the values of the fields that will be suppressed. The CMTS MUST use J.112 MAC messaging to convey to the CM the fact that suppression should be used for a particular flow, and instructs it to suppress appropriate fields given the presence or absence of UDP checksums and IP Sequence numbers.

If the MTA initiates a PATH message specifying a wildcard sender, then no contents of the PHS field can be accurately determined. The CMTS MUST specify the PHS Size so that the CM can accurately assess the resource needs of the service flow.

The same basic approach enables support of Voice Activity Detection (VAD). A CMTS may use different scheduling algorithms for flows that are using VAD, and thus needs to know which flows may be treated with VAD. The Compression_Hint parameter carried in the Tspec MUST contain the flag bit to indicate that the data flow for which this reservation is being requested may be treated with VAD.

6.2.4 UGS and UGS/AD authorization and behaviour

The GC does not specify the J.112 service flow scheduling type for DQoS flows. As such, guidelines are established regarding their use.

For a normal session:

- The CMTS should select UGS or UGS/AD service flow scheduling type for a stand-alone MTA using RSVP based upon the VAD compression hint parameter. It may provide provisioned parameters to control the decision.

For cases where a reservation covers multiple flowspecs (Least Upper Bound):

- In RSVP messaging if VAD is indicated, the CMTS should create a UGS/AD service flow scheduling type. The CMTS may also provide provisioned parameters to control the decision.

For cases where resources are to be shared among flows:

- In RSVP messaging the CMTS must use the same service flow scheduling type for all shared resources. Therefore, the MTA must request identical settings for VAD. The CMTS will reject any request to share resources if the VAD setting does not match the existing flow.

6.2.5 CMTS authorization and behaviour

The CMTS upon receiving bandwidth reservation or commitment requests containing a GateID, must perform admission control on the bandwidth request using the gate objects associated with the GateID.

Each DSA or DSC request originating from an E-MTA in support of a particular call session MUST contain a GateID in the Authorization Block, otherwise the CMTS MUST reject the request with confirmation code 24 (Authorization Failure). If a DSC request message is received which contains a GateID different from the GateID provided in the DSA request used to create the service flow,

then the CMTS MUST perform normal authorization and admission control procedures using the Gate associated with the new GateID.

If authorization and admission control succeed, the CMTS MUST associate the new GateID with the modified service flow, replace the DOCSIS Admitted Flow Timeout and Active Flow Timeout values of the associated service flow with the T7 and T8 timers of the new upstream Gate, and include those timer values in the DSC response to the MTA. In this case the CMTS MUST remove the original Gate immediately and notify the CMS via a Gate-Close with Reason-Sub-Code 0 (Normal).

The CMTS and CMS elements MUST NOT reuse a Gate previously associated with a service flow in authorizing a separate service flow. A CMTS MUST reject a reservation or commit request for a new service flow against a Gate authorizing a separate service flow with DOCSIS confirmation code 24 (Authorization Failure).

If the IPCablecom authorization module receives a bandwidth reservation request without an authorization block, the CMTS MUST reject the request with error code "*authorization rejected permanent*".

If the CMTS cannot find a gate associated with the GateID, it MUST return an error code indicating that this request has failed authorization and will be permanently rejected.

If the CMTS finds a gate associated with the GateID, then the CMTS must conduct the following authorization procedure. In order to perform admission control on DOCSIS DSx messages and to compare these messages on a parameter basis with those authorized via the GateSpec object, the CMTS must normalize QoS parameters either to layer-two or layer-three by adding or subtracting link-layer overhead. The examples provided in this Recommendation assume that normalization results in layer-three parameters by converting DOCSIS parameters to their RSVP equivalents using the methods described in 6.2.

Sender Tspec and *Reverse Sender Tspec*:

The GateSpec Bucket Depth (b), MUST be greater than or equal to the MTA requested value.

The GateSpec Bucket Rate (r), MUST be greater than or equal to the MTA requested value.

The GateSpec Maximum Datagram Size (M), MUST be greater than or equal to the MTA requested value.

The GateSpec Minimum policed unit (m), MUST be greater than or equal to the MTA requested value.

The GateSpec Peak Rate (p), MUST be greater than or equal to the MTA requested value.

Sender Rspec and *Reverse Sender Rspec*:

The GateSpec Reserved Rate (R), MUST be greater than or equal to the MTA requested value.

The GateSpec Slack Term (s), MUST be less than or equal to the MTA requested value.

Session and *Reverse Session*:

The GateSpec Protocol MUST be equivalent to the MTA requested protocol.

The GateSpec Destination Address MUST be the same as the MTA requested address, if the GateSpec contains a non-zero value. If the GateSpec contains a zero value, then this comparison MUST be omitted.

The GateSpec Destination Port MUST be the same as the MTA requested port if the GateSpec contains a non-zero value. If the GateSpec contains a zero value, then this comparison MUST be omitted.

Sender Template and *Reverse Sender Template*:

The GateSpec Source Address MUST be the same as the MTA requested address, if the GateSpec contains a non-zero value. If the GateSpec contains a zero value, then this comparison MUST be omitted.

The GateSpec Source Port MUST be the same as the MTA requested port if the GateSpec contains a non-zero value. If the GateSpec contains a zero value, then this comparison MUST be omitted.

If one of the above authorization comparisons fails for a message requesting a new service flow or altering the reservation parameters of an existing flow, then the CMTS MUST NOT honour the request by creating a new service flow or altering the parameters of the existing service flow. If the MTA requests a commit operation for a reserved flow, then the authorization MUST be carried out using the DOCSIS parameters and the method defined in DOCSIS.

6.3 Definition of additional RSVP objects

Several new RSVP objects MUST be added to the original PATH message sent by the MTA. All new objects have a class-number with the high order two bits set, which means that RSVP nodes that do not recognize these objects should forward them without modification. This clause defines the formats of the various new objects that are to be carried in RSVP messages. All objects use the encoding scheme of RSVP IETF RFC 2205.

6.3.1 Reverse-Rspec

Reverse-Rspec object: Class = 226, C-type = 1.

Length (= 24)		Class (= 226)	C-Type (= 1)
0 (a)	Reserved	4 (b)	
2 (c)	0 Reserved	3 (d)	
130 (e)	0 (f)	2 (g)	
Rate [R] (32-bit IEEE floating point number)			
Slack Term [S] (32-bit integer)			

(a) – Message format version number (0).

(b) – Overall length (4 words not including header).

(c) – Service header, service number 2 (Guaranteed).

(d) – Length of service 1 data, 3 words not including header.

(e) – Parameter ID, parameter 130 (Guaranteed Service Rspec).

(f) – Parameter 130 flags (none set).

(g) – Parameter 130 length, 2 words not including parameter header.

See IETF RFC 2210 for explanation of fields.

The Reverse-Rspec applies to data sent by the client, i.e., upstream in the J.112 network. It is included in the PATH message sent by the client, and is turned into the Forward-Rspec object in the RESV message generated by the CMTS in its role as proxy for the remote endpoint.

6.3.2 Reverse-Session

IPv4 Reverse Session object:

Length (= 12)		Class (= 226)	C-Type (= 2)
IPv4 Destination Address (4 bytes)			
Protocol ID	Flags	Destination Port	

The Reverse-Session object describes the destination information of the data stream to be received by the MTA, i.e., downstream in the J.112 network. It becomes the Session Object in the PATH message generated by the CMTS in its role as proxy for the remote endpoint.

6.3.3 Reverse-Sender-Template

IPv4 Reverse-Sender-Template object:

Length (= 12)		Class (= 226)	C-Type (= 3)
IPv4 Source Address (4 bytes)			
Reserved	Reserved	Source Port	

The Reverse-Sender-Template describes the source information of the data stream to be received by the MTA, i.e., downstream in the J.112 network. It becomes the Sender-Template object in the PATH message generated by the CMTS in its role as proxy for the remote endpoint.

6.3.4 Reverse-Sender-Tspec

Reverse-Sender-Tspec object: Same fields as Sender-Tspec described in RFC 3006.

Length (= 48)		Class (= 226)	C-Type (= 4)
0 (a)	Reserved	10 (b)	
1 (c)	0 Reserved	9 (d)	
127 (e)	0 (f)	5 (g)	
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)			
126 (h)	flags (i)	2 (j)	
Hint (assigned number) (k)			
Compression factor (32-bit integer) (l)			

(a) – Message format version number (0).

(b) – Overall length (10 words not including header).

(c) – Service header, service number 1 (default/global information).

(d) – Length of service 1 data, 9 words not including header.

(e) – Parameter ID, parameter 127 (Token_Bucket_Tspec).

(f) – Parameter 127 flags (none set).

(g) – Parameter 127 length, 5 words not including header.

(h) – Parameter ID, parameter 126 (Compression_Hint).

(i) – Parameter 126 flags (none set).

(j) – Parameter 126 length, 2 words not including header.

(k) – Hint value defined for J.112 header suppression (TBD).

0x????0001 Do not suppress UDP checksum AND do not suppress IP-Ident field nor IP-Checksum field.

0x????0002 Do not suppress UDP checksum AND suppress IP-Ident and IP-Checksum field.

0x????0003 Suppress UDP checksum AND do not suppress IP-Ident nor IP-Checksum field.

0x????0004 Suppress UDP checksum AND suppress IP-Ident field and IP-Checksum field.

NOTE – ???? = To be determined IANA number assignment for IPCablecom.

(l) – Compression factor value – The percentage reduction in packet size as a result of using J.112 header suppression. Note this varies depending on the CODEC used. See IETF RFC 2210 for explanation of fields.

The Reverse-Sender-Tspec describes the data flow to be sent by the MTA, i.e., upstream in the J.112 network. It becomes the Sender-Tspec object in the PATH message generated by the CMTS in its role as proxy for the remote endpoint.

6.3.5 Forward-Rspec

Forward-Rspec object:

Length (= 24)		Class (= 226)	C-Type (= 5)
0 (a)	Reserved	4 (b)	
2 (c)	0 Reserved	3 (d)	
130 (e)	0 (f)	2 (g)	
Rate [R] (32-bit IEEE floating point number)			
Slack Term [S] (32-bit integer)			

For definitions of these fields, see 6.3.1, Reverse-Rspec.

The Forward-Rspec applies to data flowing toward the client, i.e., downstream in the J.112 network. This object appears in a PATH message sent by the client, and the contents are incorporated into the Flowspec object in the returned RESV message.

6.3.6 Resource-ID

Resource-ID object:

Length (= 8)	Class (= 226)	C-Type (= 7)
Resource ID (32-bit integer)		

The Resource-ID object is returned in a RESV message to the MTA, and contains the identifier used for future resource changes. It is also included in PATH messages sent by the MTA in requests to share resources across multiple sessions.

6.3.7 Gate-ID

Gate-ID object:

Length (= 8)	Class (= 226)	C-Type (= 8)
Gate ID (32-bit integer)		

The Gate-ID object is included in PATH messages from the MTA to identify the proper resource authorization at the CMTS.

6.3.8 Commit-Entity

IPv4 Commit-Entity object:

Length (= 12)	Class (= 226)	C-Type (= 9)
IPv4 Destination Address (4 bytes)		
Reserved	Destination Port	

The Commit-Entity object is returned in a RESV message from the CMTS, and indicates the destination address and port number to which the MTA is to send the COMMIT message.

6.3.9 DClass

DClass object:

Length (= 8)		Class (= 225)	C-Type (= 1)
Unused	Unused	Unused	DSCP

The DClass object is returned in a RESV message from the CMTS, and indicates the DSCP that SHOULD be used by the MTA when sending data packets over this reservation to the CMTS. The use of the DClass object is described in Use and Format of the DCLASS Object with RSVP Signalling [IETF RFC 2996].

6.4 Definition of RSVP messages

This clause defines the enhanced RSVP messages that MUST be generated by the MTA and MUST be supported by the CMTS.

RSVP messages MUST be sent as "raw" IP datagrams with protocol number 46. The RSVP-PATH message MUST be sent with the RouterAlert option IETF RFC 2113 in the IP header. Each RSVP message MUST occupy exactly one IP datagram.

All RSVP messages MUST consist of a Common Header, followed by a variable number of variable-length objects. The Common Header MUST be as follows:

Version	Flags	Message Type	RSVP Checksum
Sent-TTL		(Reserved)	RSVP Message Length

Values of each field MUST be as specified in IETF RFC 2205.

Each object MUST consist of one or more 32-bit words with a one-word header, of the following format:

Length in bytes	Class-Number	C-Type
Object Contents …		

Values of each field MUST be as specified in IETF RFC 2205.

The format of the RSVP-PATH message and RSVP-RESV message compliant with this Recommendation MUST contain the following objects (items in italics are defined in this Recommendation, all others in IETF RFC 2205 and/or IETF RFC 2210). For objects not defined in this Recommendation, the object ordering rules MUST be followed according to IETF RFC 2205. There is no ordering requirement for the <Resource-ID>, <Gate-ID>, and <Commit-Entity> objects. <Reverse-Rspec> and <Downstream-Flowspec> MUST follow the <Sender-Tspec> object. Objects defined in <Downstream-Flowspec> and <Component-Item> MUST follow the order shown in their BNF below:

<PATH-Message> ::=Common-Header> [<Integrity-Object>]
<Session-Object> <RSVP-Hop> <Time-Values>
<Policy-Data> …] <Sender-Template>
Sender-Tspec> <*Reverse-Rspec*>
Downstream-Flowspec> [<*Resource-ID*>]
Gate-ID>

<*Downstream-Flowspec*> *::=* <*Reverse-Session*> <*Reverse-Sender-Template*>
<*Reverse-Sender-Tspec*><*Forward-Rspec*>

<RESV-Message> ::= <Common-Header> [<Integrity-Object>]
<Session-Object> <RSVP-Hop> [<DClass>]
Time-Values> [<RESV-Confirm>] [<Scope>]
<Policy-Data> …] <*Resource-ID*>
Commit-Entity> <Style> <Flowspec>
Filter-Spec>

The various components of these messages are described in the following subclauses.

6.4.1 Message objects for upstream reservation

A standard RSVP-PATH message contains, at a minimum, the following objects:

<Session> <RSVP-Hop> <Time-Values> <Sender-Template> <Sender-Tspec>

However, in the segmented model, it is necessary to get all the information to the CMTS that would enable it to make a bidirectional reservation on the J.112 link. It is also necessary to enable it to send an RSVP-RESV towards the MTA. A standard RSVP-RESV message contains, at a minimum, the following objects:

<Session> <RSVP-Hop> <Time-Values> <Style> <Flowspec> <Filter-Spec>

The CMTS MUST generate such a message towards the MTA after receiving an RSVP-PATH message from the MTA. The only object here that is not derivable from the RSVP-PATH or local information is the Flowspec. The Filter-Spec, which consists of the IP address and source port to be used by the MTA, is derived from the Sender-Template in the PATH. Almost everything in the Flowspec can be derived from the Sender-Tspec in the PATH message. The exceptions to this are the values of R (reserved rate) and S (slack), which together constitute the Rspec. Thus, the MTA provides a suitable Rspec, containing R and S for guaranteed service, which is encoded as in IETF RFC 2210. This is enclosed in a Reverse-Rspec object, which is described in 6.3.1.

6.4.2 Message objects for downstream reservation

The MTA MUST provide enough information to allow the CMTS to construct an RSVP-PATH message for the downstream data flow given that it has just received an RSVP-PATH message for the upstream data flow. This means the MTA MUST provide the following objects that relate to the downstream (CMTS→MTA) data flow:

<Session> <Sender-Template> <Sender-Tspec>

These objects have their normal RSVP definitions, and apply to the simplex data stream that will flow from the far endpoint to the MTA. In the RSVP-PATH message sent by the MTA, they are assigned new object codes (as noted above) and new names (Reverse-session, Reverse-sender-template, Reverse-Sender-Tspec). The Reverse-Session-Object MUST contain the IP address of the MTA, the protocol type, and the port (if applicable) on which it will receive data for this flow. The Reverse-Sender-Template MUST contain the IP address of the far endpoint, or zeroes if the source is intended as a wildcard. The Reverse-Sender-Template MUST contain the port number, if applicable and known, otherwise zero. The Reverse-Sender-Tspec MUST contain the Tspec information that describes the data flow from the far endpoint. The CMTS MUST use its own address as the RSVP-Hop and choose a value for Time-Values that indicates how often it will refresh the RSVP-PATH message. Even if the CMTS does not need to generate the RSVP-PATH message to send it to the MTA, this information is necessary to enable it to establish a reservation and create packet classifiers in the downstream direction.

Given the information described above, the one additional piece of information that the CMTS requires to make a reservation in the downstream direction is an Rspec. Again, it is assigned a new object number and name, Forward-Rspec. It contains the same information elements and is encoded in the same way as a conventional Rspec.

Note that a Forward-Rspec applies to data that is flowing towards the MTA, which means that it is sent by the MTA in the same direction as the RSVP-RESV that would normally carry this information. It is provided in the RSVP-PATH message simply as an optimization to reduce set-up latency. A Reverse-Rspec is sent by the MTA in the opposite direction to the RSVP-RESV that would normally carry this information.

6.5 Reservation operation

This clause describes the required behaviour of the MTA and CMTS to cooperatively perform resource reservations.

For the purposes of this discussion, the endpoint that is in direct communication with the CMTS is referred to as the client, and the other endpoint of the session is referred to as the far endpoint. We make no assumptions about what types of devices these might be (gateways, PCs, embedded clients). We assume that the client uses RSVP to communicate QoS requests to the CMTS, and we make no assumptions about the capabilities of the far endpoint. The data flow from client to CMTS is referred to as upstream, and the flow from CMTS to client is referred to as downstream.

6.5.1 Reservation establishment

RSVP operation under the segmented model is as follows:

The client MUST send a RSVP-PATH message towards the far endpoint of the session, which MUST be intercepted by the CMTS. This initiates the process of reserving both upstream and downstream bandwidth. The RSVP-PATH MUST carry information about both upstream (i.e., Reverse-Rspec) and downstream (i.e., Reverse-Sender-Tspec, Forward-Rspec) resource requirements in the case where reservations are required in both directions.

The CMTS MUST verify that the amount of resources requested is within the authorized amount for this session and that it has sufficient local resources to accommodate the reservation. It then reserves upstream and downstream resources and MUST perform the J.112 MAC-level messaging to allocate appropriate resources on the J.112 link.

The CMTS MUST establish classifiers for the upstream and downstream flows. The upstream classifier MUST contain the client's source IP address and port number from the Sender Template object. The upstream classifier MUST contain the protocol type, destination IP address and port number from the Session Object. If the Reverse-Sender-Template Object is present, and contains an address other than 0.0.0.0, then the downstream classifier MUST contain this address as the source IP address. If the Reverse-Sender-Template is present, and contains a port number other than 0, then the downstream classifier MUST contain this value as the source port. The downstream classifier MUST contain the protocol type, destination IP address, and port number from the Reverse Session Object.

The CMTS MUST perform any backbone resource reservation necessary, based on the provisioned algorithm defined for the particular backbone configuration.

If the access and backbone reservations succeed, the CMTS MUST send a RSVP-RESV to the client. The contents of the RSVP-RESV MUST be derived from the RSVP-PATH: the Session-Object is copied from the RSVP-PATH, Style is set to Fixed-Filter, Flowspec is formed from the Sender-Tspec and Forward-Rspec, Filter-Spec is set from the Sender-Template, and the Resource-ID is generated, containing the Resource-ID assigned to the allocated resources. The Commit-Entity Object MUST be included, and contain the address of the CMTS and port number on which the CMTS will accept the COMMIT message (as described in 6.6). The DCLASS object SHOULD be included and value set based on the Diffserv Code Point field of the gate.

If the address of the previous hop differs from the Source Address of the RSVP-PATH message, then the CMTS MUST generate a RSVP-PATH for downstream reservations. The contents of the RSVP-PATH MUST be derived from the RSVP-PATH received from the client. The

Session-Object MUST be obtained from the Reverse-Session-Object in the RSVP-PATH message. If the address contained in the Reverse-Sender-Template is 0.0.0.0, or the port number is 0, then the Sender-Tspec and Sender-Template are not sent in the RSVP-PATH. Otherwise, the Sender-Tspec is obtained from the Reverse-Sender-Tspec, the Forward-Rspec is obtained from the Reverse-Rspec, and the Sender-Template is obtained from the Reverse-Sender-Template. The Resource-ID object is generated, and contains the Resource-ID assigned to the allocated resources. The MTA MAY use the Reverse-Sender-Tspec it sent in the RSVP-PATH message in calculating the Filter-Spec returned in its RSVP-RESV reply, or MAY generate a Wildcard-Filter reply.

On receipt of the RSVP-RESV message, the client knows that necessary resources have been reserved. At this point, in the case of a successful reservation, the client knows that it has a reservation in both directions, and can proceed with the call signalling to ring the phone at the far end.

If the reservation does not succeed, the CMTS MUST send a RSVP-PATH-ERR message to the client, indicating why the reservation failed (e.g., lack of authorization, insufficient resources, etc.). If the reservation failed for policy reasons, the RSVP-PATH-ERR message MUST contain a RSVP-ERROR-SPEC object with the following Error Codes and Error Values:

- Error Code = 2 (Policy Control Failure), Error Value = 3 (Generic Policy Rejection) is returned if the RSVP-PATH did not contain a Gate-ID object or the Gate-ID object did not match any gates known to the CMTS.
- Error Code = 1 (Admission Control Failure), Error Value = 2 (Requested bandwidth unavailable) is returned if the RSVP-PATH was rejected because there was no more resources that are available for the priority level of the gate. In these cases, the MTA MAY take special action indicating the specific error to the user. If the RSVP-PATH failed for non-policy reasons, it MUST contain a RSVP-ERROR-SPEC object with an Error Code and Error Value as defined in Appendix B of IETF RFC 2205.

The sender of a RSVP-PATH (MTA or CMTS) is responsible for reliably installing the reservation. When the sender transmits a RSVP-PATH, it MUST receive an RSVP-RESV or RSVP-PATH-ERR message within a configured timeout interval of Timer T3 (see Annex A).

Whenever an MTA or CMTS transmits an RSVP message that requires an acknowledgement, the sender MUST include an RSVP-MESSAGE-ID object in that message, and the ACK_Desired flag of the RSVP-MESSAGE-ID object MUST be set. The MTA and CMTS MUST set the Refresh-Reduction-Capable flag in the common header of every RSVP message. When the MTA or CMTS receives an RSVP message with an RSVP-MESSAGE-ID object, it MUST respond with an RSVP message that contains an RSVP-MESSAGE-ACK or RSVP-MESSAGE-NACK object. The RSVP-MESSAGE-(N)ACK object MAY be piggy-backed onto standard RSVP messages, but MAY be transmitted in an RSVP-ACK message if the receiver of the RSVP-MESSAGE-ID object has no other RSVP message to send at the time. For example, the CMTS SHOULD NOT delay processing of a received RSVP-PATH message, but if it chooses to delay, it MUST reply immediately with an RSVP-ACK message, to be followed by a RSVP-RESV message later.

RSVP-ACK messages carry one or more RSVP-MESSAGE-(N)ACK objects. They MUST NOT contain any other RSVP objects except an optional RSVP-INTEGRITY object. When included, an RSVP-MESSAGE-(N)ACK object MUST be the first object in the message, unless an RSVP-INTEGRITY object is present (in which case, the RSVP-MESSAGE-(N)ACK object MUST immediately follow the RSVP-INTEGRITY object). The MTA or CMTS MAY use RSVP-INTEGRITY objects.

The use of RSVP-MESSAGE-ID and RSVP-MESSAGE-(N)ACK objects can be used to ensure reliable RSVP message delivery in the face of network loss. Since the MTA or CMTS sets the ACK_Desired flag, it MUST retransmit unacknowledged messages at a more rapid interval than the standard RSVP refresh interval until the message is acknowledged or until an interval of Timer T3

(see Annex A) elapses. A rapid retransmit rate based on well-known exponential back-off functions MUST be used. An initial retransmit timeout of Timer T6 (see Annex A) MUST be used, with a power of 2 back-off. The rapid retransmit process ends when either an RSVP-MESSAGE-(N)ACK object is received or Timer T3 expires. If RSVP-PATH sender does not receive a RSVP-RESV, RSVP-PATH-ERROR, or RSVP-MESSAGE-(N)ACK before the next retransmit, it MUST assume either its original RSVP-PATH or the response from the other end was lost and re-sends the RSVP-PATH. Since all RSVP messages are idempotent, no duplications of reservations will occur.

In IPCablecom, only RSVP-PATH messages MUST include RSVP-MESSAGE-ID objects with the ACK_Desired flag set. RSVP-MESSAGE-ID objects MAY be used in other RSVP messages.

RSVP-MESSAGE-IDs are used on a per-RSVP-hop basis. Each RSVP-capable hop in the path that supports refresh reduction does its own fast retransmit until it sees an acknowledgement from the next upstream node. So if a stand-alone MTA behind an RSVP-capable CM receives an RSVP-MESSAGE-ACK object from the CM for an RSVP-PATH and the CM is waiting for an RSVP-MESSAGE-ACK from the CMTS for the RSVP-PATH, the CM will do the fast retransmit while the stand-alone MTA waits for its normal (30 s) RSVP-PATH refresh Timer To expire. (The MTA no longer does fast retransmit because it got an acknowledgement.) If an RSVP-capable CM gives up its fast retransmit, it will send back an RSVP-PATH-ERROR to the stand-alone MTA. This way, retransmits do not affect the entire path, just the loss-prone hops.

Reliable message delivery for RSVP messages is defined and described more completely in IETF RFC 2961. IETF RFC 2961 also includes mechanisms to reduce the number of RSVP signalling messages required to refresh RSVP state. The MTA and CMTS implementations of IETF RFC 2961:

- MUST enable the Refresh-Reduction-Capable bit in the RSVP header;
- MUST support the MESSAGE_ID extension;
- MUST support the Summary Refresh extension for unicast sessions;
- MAY support the Summary Refresh extension for multicast sessions;
- MUST support reception of Bundle messages;
- MAY support transmission of Bundle messages.

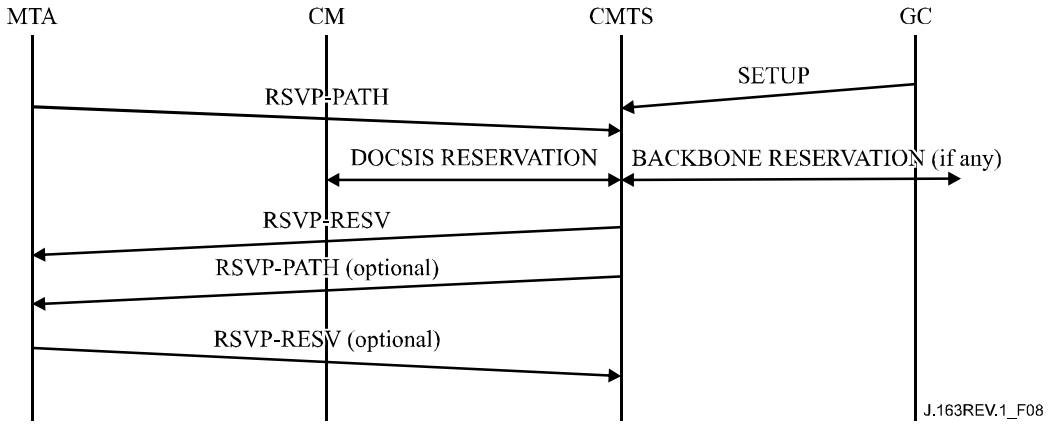


Figure 8/J.163 – Reservation establishment

The CMTS MUST enforce upstream packet classification filters for J.112 Flows. That is, the CMTS MUST discard upstream packets, which do not match the set of upstream packet classifiers for the J.112 Flow. Upstream packet classification filtering is an optional CMTS requirement in J.112 networks. This Recommendation requires its implementation for J.112 Flows used to carry IPCablecom media streams. If a CMTS chooses to enforce upstream classification filters only on

the J.112 Flows, and not on other flows, it is a CMTS vendor-specific decision as to how the particular J.112 Flows are determined.

6.5.2 Reservation change

In addition to establishing a reservation for some amount of resources, it may be necessary to change the resources allocated. Resource usage may need to be increased or decreased. RSVP handles changes in resource usage by changes in the FLOWSPEC object of a RSVP-RESV message and/or a change in the Sender-Tspec in a RSVP-PATH message. A reservation change MUST follow the same series of steps as the establishment of a new reservation. Admission control SHOULD always succeed for a session, which is changing its resource requirements in a way that does not cause an increase in any dimension relative to the resources previously reserved. Because resources are described by multi-dimensional vectors, a change in reservation that increased resources in one direction and decreased them in another MUST pass through admission control. Note that in order to pass admission control, the resources MUST be within the amount of authorized resources for the session and also within the amount of resources available to the CMTS.

In the event that an existing reservation is pre-empted because a session with a higher priority gate must be established in the presence of insufficient bandwidth, then the CMTS MUST send a RSVP-PATH-ERR and/or RSVP-RESV-ERR message for the session that is being pre-empted. This message SHOULD be sent as soon as possible. In response, the MTA SHOULD tear down the reservation and MAY notify the user of the pre-emption (e.g., play a special tone to phone user). The RSVP-PATH-ERR (or RSVP-RESV-ERR) message in this case MUST contain a RSVP-ERROR-SPEC object with an Error Code of 2 (Policy Control Failure) and an Error Value of 5 (Flow was pre-empted).

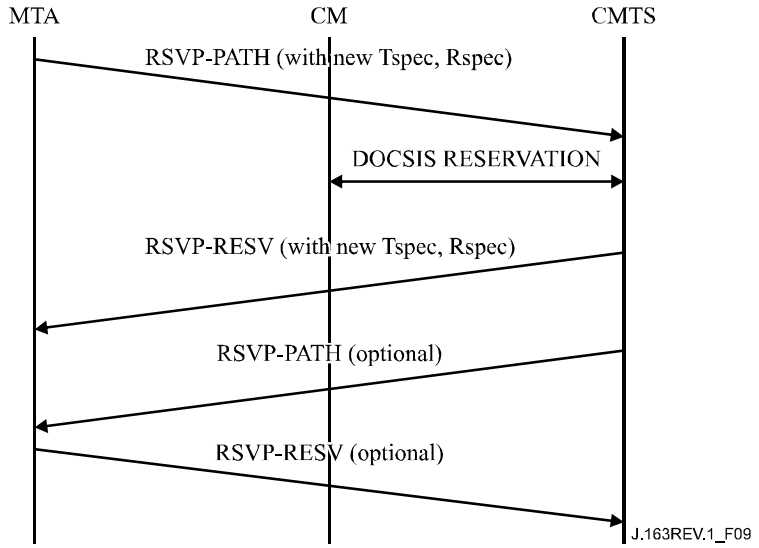


Figure 9/J.163 – Reservation change

6.5.3 Reservation deletion

RSVP provides two messages for the explicit removal of Path and Reservation state, the RSVP-PATH-TEAR and RSVP-RESV-TEAR messages. To delete a reservation at the CMTS, the MTA SHOULD send a RSVP-PATH-TEAR message. To delete a reservation from RSVP-capable devices between the MTA and the CMTS, the MTA MAY send a RSVP-RESV-TEAR message. The format of these messages MUST be in accordance with IETF RFC 2205, and MUST include both the Session-Object and Sender-Template to enable the CMTS to identify the proper gate.

If Path state and Reservation state are not periodically refreshed, they MUST time out. This is appropriate when a MTA crashes, for example. More details of refresh mechanisms appear in 6.5.4.

The CMTS MUST respond to a received RSVP-PATH-TEAR by sending a RSVP-RESV-TEAR to the MTA. The format of these messages MUST be as given in IETF RFC 2205.

RSVP version 1 provides no means to ensure the reliable delivery of RSVP-PATH-TEAR and RSVP-RESV-TEAR messages, on the assumption that the state which they aim to delete will time out anyway. However, to avoid any delay in teardown (which causes short-term resource wastage and may cause overbilling), the message reliability extension to RSVP described in [IETF RFC 3209] may be used.

6.5.4 Reservation maintenance

RSVP has a soft-state model, in that reservation state times out if not periodically refreshed. This characteristic is retained in the segmented model described here. Since the entire reservation process in this model is initiated by the MTA, the MTA MUST periodically refresh all RSVP state information. The MTA MUST send RSVP-PATH messages as described in 6.5.1 within the time interval given by the CMTS in the RSVP-RESV Time-Values Object. The CMTS MUST generate RSVP-RESV messages towards the MTA on receipt of the RSVP-PATH (and a RSVP-PATH message as well if RSVP capable nodes have been detected as described in 6.5.1). This retains the soft state nature of RSVP, which retains its resiliency in the face of routing changes and node failures.

The MTA (or CMTS) MAY also implement RSVP Summary Refresh as another way to conserve upstream bandwidth when refreshing reservation state. It allows RSVP-capable nodes to "compress" their Path (or Resv) states for multiple reservations into single message. RSVP Refresh Overhead Reduction Extensions [IETF RFC 2961] describes summary refresh as follows:

"The summary refresh extension enables the refreshing of RSVP state without the transmission of standard Path or Resv messages. The benefits of the described extension are that it reduces the amount of information that must be transmitted and processed in order to maintain RSVP state synchronization. Importantly, the described extension preserves RSVP's ability to handle non-RSVP next hops and to adjust to changes in routing. This extension cannot be used with Path or Resv messages that contain any change from previously transmitted messages, i.e., are trigger messages.

The summary refresh extension builds on the previously defined MESSAGE_ID extension. Only state that was previously advertised in Path and Resv messages containing MESSAGE_ID objects can be refreshed via the summary refresh extension.

The summary refresh extension uses the objects and the ACK message previously defined as part of the MESSAGE_ID extension, and a new Srefresh message. The new message carries a list of Message_Identifier fields corresponding to the Path and Resv trigger messages that established the state. The Message_Identifier fields are carried in one of three Srefresh related objects. The three objects are the MESSAGE_ID LIST object, the MESSAGE_ID SRC_LIST object, and the MESSAGE_ID MCAST_LIST object.

The MESSAGE_ID LIST object is used to refresh all Resv state, and Path state of unicast sessions. It is made up of a list of Message_Identifier fields that were originally advertised in MESSAGE_ID objects. The other two objects are used to refresh Path state of multicast sessions. A node receiving a summary refresh for multicast path state will at times need source and group information. These two objects provide this information. The objects differ in the information they contain and how they are sent. Both carry Message_Identifier fields and corresponding source IP addresses. The MESSAGE_ID SRC_LIST is sent in messages addressed to the session's multicast IP address. The MESSAGE_ID MCAST_LIST object adds the group address and is sent in messages addressed to the RSVP next hop.

The MESSAGE_ID MCAST_LIST is normally used on point-to-point links.

An RSVP node receiving an Srefresh message, matches each listed Message_Identifier field with installed Path or Resv state. All matching state is updated as if a normal RSVP refresh message has been received. If matching state cannot be found, then the Srefresh message sender is notified via a refresh NACK.

A refresh NACK is sent via the MESSAGE_ID_NACK object. As described in the previous clause, the rules for sending a MESSAGE_ID_NACK object are the same as for sending a MESSAGE_ID_ACK object. This includes sending MESSAGE_ID_NACK object both piggy-backed in unrelated RSVP messages or in RSVP ACK messages."

Complete details on how summary refresh works can be found in section 5 of RSVP Refresh Overhead Reduction Extensions [IETF RFC 2961].

6.6 Definition of Commit messages

This clause defines the Commit messages that MUST be generated by the MTA and MUST be supported by the CMTS.

Commit messages MUST be sent as UDP/IP datagrams with protocol number 17 (UDP). Each Commit message MUST occupy exactly one UDP/IP datagram. The destination IP address and port number in the UDP header MUST be as specified from the Commit-Entity Object returned in the RSVP-RESV message. The source port number MUST be the port on which the MTA will accept the acknowledgement message.

The Commit messages MUST consist of a Common-Header, followed by a variable number of variable-length objects. The Common Header MUST be as follows:

Version	Flags	Message Type	Message Checksum
Sent-TTL		(Reserved)	Message Length

Values of each field MUST be as specified in IETF RFC 2205. Message types MUST be as follows:

COMMIT	240
COMMIT-ACK	241
COMMIT-ERR	242

Each object MUST consist of one or more 32-bit words, with a one-word header of the following format:

Length in bytes	Class-Number	C-Type
Object Contents …		

Values of each field MUST be as specified in IETF RFC 2205.

The format of the COMMIT message and COMMIT-ACK message compliant with this Recommendation MUST be as follows (items in italics are defined in 6.3, all others in IETF RFC 2205 and/or IETF RFC 2210):

<COMMIT-Message> ::= <Common-Header> <Session>
Sender-Template> <*Gate-ID*>
<Flowspec>] [<*Downstream-Flowspec*>]
<COMMIT-ACK-Message> ::= <Common-Header> <Session>
Sender-Template><*Gate-ID*>
<COMMIT-ERR-Message> ::= <Common-Header> <Session>
<Sender-Template><*Gate-ID*><Error-Spec>

See 6.4, Definition of RSVP Messages for the BNF definition of <*Downstream-Flowspec*>.

The Session and Sender-Template objects identify the sender and destination IP addresses and ports, and MUST be present. The Committed resources MAY be less than the total reserved resources (especially in a call-waiting or codec-change scenario), so that a Commit message MAY also contain a <Flowspec> object for each direction of the session. This provides a mechanism by which the size of the committed resources can be modified up or down as long as the amount of resources committed does not exceed the reserved resources. Note that a set of resources MAY be put on hold (frozen) by lowering the committed resources to zero while leaving the reserved resources in place. If either flowspec is omitted, the CMTS MUST set the amount of committed resource in that direction to equal to the amount of reserved resources.

6.7 Commit operations

A significant aspect of the Dynamic QoS model is that reservation is a two-phase process, with a Commit phase following the Reserve phase. Clause 6.5 above described the Reserve phase, while this clause describes the Commit Phase and its relationship to the Reserve phase.

A conformant CMTS MUST perform all admission control and resource allocation functions on receipt of the original RSVP-PATH message, but MUST NOT allow access to those resources by the MTA until a COMMIT message is received, unless told otherwise in the GATE-SET parameters.

To perform a COMMIT the MTA MUST send a unicast message to the CMTS. This is desirable because the Commit phase only involves a MTA and a gate. The MTA learns the CMTS address and port number from the Commit-Entity object in the RSVP-RESV message.

Note that a COMMIT message differs in an important way from a standard RSVP message. It is sent directly from the MTA to the CMTS rather than hop-by-hop as an RSVP message would be. However, it contains objects that are syntactically the same as RSVP objects.

The CMTS MUST verify the value of Gate-ID, and verify the contents of the Session and Sender-Template objects match the previous reservation with the same value of Gate-ID, and that the Reverse-Session and Reverse-Sender-Template, if present, match the previous reservation with the same value of Gate-ID. The CMTS MUST acknowledge the receipt of a COMMIT with a COMMIT-ACK message or a COMMIT-ERR message.

A COMMIT-ACK is sent after the successful completion of J.112 DSC message exchange between the CMTS and the CM. If the J.112 messaging fails, a COMMIT-ERR is sent instead. When an MTA does not receive the acknowledgement within a timeout interval of Timer T4 (see Annex A), the MTA MUST resend the COMMIT, up to a limit of seven attempts.

If the MTA desires to change the amount of committed resources within the reserved envelope, another COMMIT/COMMIT-ACK sequence is REQUIRED.

If the MTA desires to change the amount of reserved resources, then the RSVP-PATH/RSVP-RESV exchange MUST be repeated.

7 Embedded MTA to CM QoS Protocol (pkt-q1)

Rather than using the pkt-q3 interface as described in clause 6, an embedded MTA MAY dynamically reserve local QoS resources using only mechanisms defined in ITU-T Rec. J.112. Using this alternate approach, an embedded MTA directly signals for the local access QoS using the MAC Control Service interface defined in Annex E to Annex B/J.112. As opposed to clause 6, the QoS signalling across the J.112 interface (pkt-q2 interface) is initiated by the CM instead of the CMTS. All other interfaces remain unchanged. An illustrative example of this approach is given in Appendices VII and VIII.

An embedded MTA signals its session level QoS requirements in signalling protocols (SIP IETF RFC 2543 and ITU-T Rec. J.162). Once the embedded MTA determines that QoS resources need to be reserved or committed, the MTA MUST initiate J.112 Dynamic Service Flow signalling to cause the creation, change, and/or deletion of Service Flow(s) and the allocation of J.112 resources. Whether the session is originated by the embedded MTA or by a peer or network node, the MTA passes the QoS requirements to the ITU-T Rec. J.112 MAC via the MAC Control Service Interface. This results in the creation or modification of the necessary Service Flow(s) for the session using the Dynamic Service Flow messaging mechanisms of ITU-T Rec. J.112. The clauses that follow discuss the MTA's mapping of session level QoS requirements into those of ITU-T Rec. J.112, the J.112 support for two-phase reserve/commit, and the use of the J.112 MAC Control Service Interface.

7.1 Mapping Flowspecs into J.112 QoS parameters

For a detailed description of the DOCSIS parameter mapping process to be used in establishing and maintaining both upstream and downstream service flows, please refer to 6.2.3. The MTA MUST use the requirements defined in that clause for mapping session level QoS requirements into DOCSIS QoS parameters.

As a supplement to these requirements, embedded MTAs MUST include their own send (i.e., upstream source) and receive (i.e., downstream destination) addresses and ports in all classifier TLVs provided via DSx messaging. Far-end addresses and receive ports MAY be wildcarded if the far-end SDP has not been provided and values have not been provided via LCO. If these values are provided in either format, they MUST be included in the classifier TLVs. Far-end source ports MUST in all cases be wildcarded since this parameter is not communicated via SDP.

It should be noted that the examples included in clause 8 do include the overhead associated with the DOCSIS BPI+ extended header, as mandated in the Security specification (ITU-T Rec. J.170). If BPI+ is disabled (e.g., for testing purposes) the values provided in these examples should be updated appropriately by subtracting five bytes of link-layer overhead from the upstream Grant Size calculation.

7.2 J.112 support for resource reservation

In ITU-T Rec. J.112 there is no defined way of passing authorization information from the CM to the *Authorization Module* within the CMTS. The Authorization Module is a logical function of the CMTS defined in ITU-T Rec. J.112. This Recommendation utilizes a new J.112 TLV which passes an Authorization Block consisting of an arbitrary string of length *n* to the CMTS to be interpreted and processed only by the Authorization Module.

The DQoS model is one in which each session is authorized. The authorization of each session uses a handle given to both the CMTS and to the MTA, which is used to match requests with authorizations. This handle is the Gate-ID. Upon receiving call signalling information, the MTA passes the Gate-ID to the CMTS using the AuthBlock TLV contained in a DSA/DSC message.

An example of the use of the Authorization Block is found as part of the DSA-REQ messages in Appendix VII.

7.2.1 Two-phase QoS Reservation/Commit

A Service Flow has three associate sets of Quality of Service Parameters, referred to as the Provisioned, Admitted, or Active QoS Parameter Set. The relationship between these is identical to the description of Authorized, Reserved, and Committed resources given in 5.7.4.

The Reserve and Commit operations are both performed by the use of J.112 Dynamic Service messages, by changing the values of the AdmittedQoSParameterSet and ActiveQoSParameterSet of the Service Flow. In a Dynamic Service Addition (DSA) or Dynamic Service Change (DSC)

message, Reserve is accomplished by including, in the Upstream Service Flow Encodings or Downstream Service Flow Encodings, the QoSParameterSetType TLV with value set to Admitted (value 2). Similarly, Commit is accomplished by setting the QoSParameterSetType TLV to Active (value 4) or Admitted+Active (value 6).

DSA and DSC exchanges between the CM and CMTS are three-way handshakes, consisting of a request message followed by a response followed by an acknowledgement. This is illustrated in Figure 10.

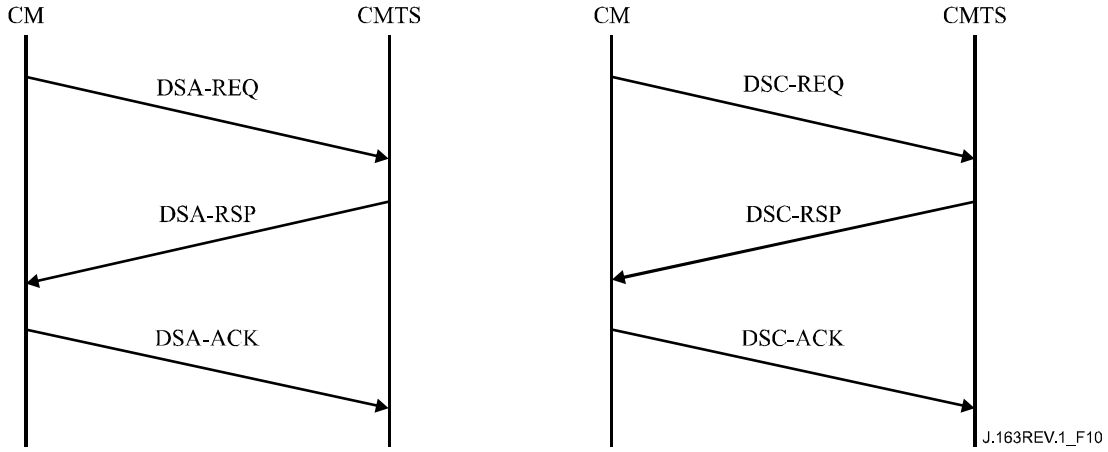


Figure 10/J.163 – DSA and DSC exchanges between CM and CMTS

For example, the following DSA-REQ message causes the Upstream and Downstream Service Flows to be admitted, meaning the QoS resources to be used in the J.112 network are reserved.

DSA-REQ

TransactionID		1
UpstreamServiceFlow	ServiceFlowReference	1
	QoSParameterSetType	Admitted (2)
	ServiceFlowScheduling	UGS (6)
	NominalGrantInterval	10 ms
	ToleratedGrantJitter	2 ms
	GrantsPerInterval	1
	UnsolicitedGrantSize	222
DownstreamServiceFlow	ServiceFlowReference	2
	QoSParameterSetType	Admitted (2)
	TrafficPriority	3
	MaximumSustainedRate	12000

As a further example, the following DSC-REQ message causes the Service Flow to be activated, meaning the QoS resources used in the J.112 network are committed.

DSC-REQ

TransactionID		1
UpstreamServiceFlow	ServiceFlowID	10288
	QoSParameterSetType	Admitted + Active (6)
	ServiceFlowScheduling	UGS (6)
	NominalGrantInterval	10 ms
	ToleratedGrantJitter	2 ms
	GrantsPerInterval	1
	UnsolicitedGrantSize	222
DownstreamServiceFlow	ServiceFlowID	10289
	QoSParameterSetType	Admitted + Active (6)
	TrafficPriority	3
	MaximumSustainedRate	12000

Specification of Admitted and Activated QoS parameter sets by the MTA is via the MAC_CREATE_SERVICE_FLOW.request and MAC_CHANGE_SERVICE_FLOW.request. By the time a Service Flow is admitted, it typically has associated classifier(s). See Appendix VII for further examples.

7.2.2 Reservation with multiple service flow specifications

There are various situations in which a reservation needs to cover a range of possible specifications. For example, some applications desire to create a reservation which can handle a switch from one flow specification to another mid-session without having to pass through admission control at each switch-over time. In order for the ActiveQoSParameterSet of a Service Flow to vary during a session, a suitable AuthorizedQoSParameterSet needs to be specified through policies at the Gate Controller. This is accomplished through the use of the least-upper-bound approach (see 6.2.1). In accordance with 6.2.4, the Least Upper Bound of flows with two different J.112 Scheduling Types is null. Refer to ITU-T Rec. J.112 for information on calculating the least-upper-bound for DSx message flows.

7.2.3 Reservation maintenance

Whereas RSVP has a soft-state model as described in 6.5.4, ITU-T Rec. J.112 provides only a timeout mechanism across the J.112 interface. The Service Flow QoS parameters "Timeout for Active QoS Parameters" and "Timeout for Admitted QoS Parameters" allow a session to be terminated and its resources released due to inactivity.

The TimeoutForActiveQoSParameters is intended to recover resources allocated to CMs that die, crash, or otherwise lose their connectivity to the cable network. Normal transmission of data packets on the service flow is sufficient to prevent this recovery action.

If the DOCSIS Active Timeout expires at the CMTS for a service flow that is authorized via a gate (i.e., a PacketCable service flow), then the CMTS will delete all service flows that are associated with the gate using a DOCSIS DSD request. The CMTS when informing the GC about the gate closure will specify "Timer T8 expiration; Service Flow inactivity at the upstream direction".

If the MTA is performing Voice Activity Detection, with a service flow scheduling type of UGS/AD, and the CMTS is actively monitoring the upstream flow for activity, then during extended silence periods the MTA MUST either send periodic data packets on the service flow or refresh the active timer via DSC messaging. The TimeoutForAdmittedQoSParameters is intended to recover resources that are reserved by a CM but not committed. In typical cases, the committed parameters will be identical to the reserved parameters, and this will not be a problem. When the

commitment is for less than the reservation, it is necessary to periodically reset the CMTS timer. This is accomplished by performing a DSC-REQ operation that reserves the same resources as before.

7.2.4 Support for dynamic binding of resources

Dynamic binding of resources, as required in 5.7.7 and described in 6.1.4, is accomplished in ITU-T Rec. J.112 through the use of the authorization block TLV.

The CMTS MUST include the Resource-ID in the authorization block TLV for the DSA-RSP message that it sends to the client. The client MAY include the Resource-ID in subsequent DOCSIS messages that apply to the resources in question. Most importantly, if the client wishes to establish a new session, and reuse the resources of an existing session, it MUST include the Resource-ID associated with the old session in the DSA-REQ message it sends to the CMTS.

7.2.5 QoS parameter mapping for authorization

The Gate identified by the GateID is parameterized by RSVP objects (both Flowspec and Tspec) for each direction. The authorization module in the CMTS must convert the DOCSIS QoS Parameters into RSVP Objects using the rules defined below:

The parameters *RSVP Bucket Depth* (b), *RSVP Maximum Datagram Size* (M), and *RSVP Minimum Policed Unit* (m) MUST be set to *DOCSIS Unsolicited Grant Size* minus the DOCSIS upstream UGS overhead[7] for upstream direction and *DOCSIS Assumed Minimum Reserved Rate Packet Size* minus the DOCSIS downstream overhead[8] for the downstream direction.

For the downstream, the parameters *RSVP Bucket Rate* (r), and *RSVP Peak Rate* (p) MUST be calculated by converting the *DOCSIS Maximum Sustained Rate* to layer 3 terms by dividing it by the *DOCSIS Assumed Minimum Reserved Rate Packet Size* and then multiplying the result with the previously calculated *RSVP Maximum Datagram Size.* For the upstream, the parameters *RSVP Bucket Rate* (r), and *RSVP Peak Rate* (p) MUST be set equal to the *DOCSIS Nominal Grant Interval* multiplied by the *Unsolicited Grant Size*.

For the downstream, the parameter *RSVP Reserved Rate* (R) MUST be calculated by converting the *DOCSIS Maximum Reserved Traffic Rate* to layer 3 terms by dividing it by the *DOCSIS Assumed Minimum Reserved Rate Packet Size* and then multiplying the result with the previously calculated *RSVP Minimum Policed Unit.* For the upstream, parameter *RSVP Reserved Rate* (R) MUST be set equal to the *DOCSIS Nominal Grant Interval* multiplied by the *Unsolicited Grant Size*.

The *RSVP Slack Term* MUST be set to *DOCSIS Tolerated Grant Jitter* for the upstream. The RSVP Slack Term MUST be set to zero for the downstream flow, indicating that this parameter will not be specified by the MTA.

The *RSVP Protocol* MUST be set to the *DOCSIS IP Protocol*.

The *RSVP Destination Address* MUST be set to *DOCSIS IP Destination Address.* If this parameter is omitted, the value MUST be set to zero.

---

[7] The overhead should include Ethernet header overhead of 18 bytes (6 bytes for source address, 6 bytes for destination address, 2 bytes for length, and 4 bytes for CRC). The value also incorporates DOCSIS MAC layer overhead, including the DOCSIS base header (6 bytes), the UGS extended header (3 bytes), and the BPI+ extended header (5 bytes). If payload header suppression (PHS) is active, then the number of suppressed bytes must be added to the DOCSIS Unsolicited Grant Size.

[8] The DOCSIS MAC layer overhead is 18 bytes (6 bytes for source address, 6 bytes for destination address, 2 bytes for length, and 4 bytes for CRC). If PHS is used on the downstream, then the number of suppressed bytes must be subtracted from the *DOCSIS Assumed Minimum Reserved Rate Packet Size.*

The *RSVP Destination Port* MUST be set to *DOCSIS IP Destination Port Start.* If this parameter is omitted, then the value MUST be set to zero.

The *RSVP Source Address* MUST be set to *DOCSIS IP Source Address*. If this parameter is omitted, the value MUST be set to zero.

The *RSVP Source Port* MUST be set to *DOCSIS IP Source Port Start*. If this parameter is omitted, then the value MUST be set to zero.

The resulting converted RSVP objects must then be verified against the corresponding Gate using the following rules:

All the requested parameters of *RSVP Flowspec* and *RSVP Slack Term* MUST be less than or equal to Gate specified values.

All the requested parameters of *RSVP Tspec* MUST be equal to the Gate-specified values, except for the case that the Gate has a value of zero, in which case the corresponding requested parameter MUST NOT be verified.

If verification is successful, then the CMTS MUST continue to process the request. If verification is not successful, then the CMTS MUST reject the request permanently due to authorization failure.

For example, assuming a G.711 codec, framing at 20 ms, with a 2-byte RTP-S MAC, and BPI+ enabled:

G.711 @ 20 ms
64 kbit/s nominal bit rate
8 kbyte/s nominal byte rate

20 ms framing rate = 50 packets/second
8 kbyte/s / 50 = 160 bytes per packet of payload

42 bytes of IP/UDP/RTP header
160 + 42 = 202 bytes per packet total
202 × 50 = 10.1 kbyte/s actual byte rate
10.1 × 8 = 80.8 kbit/s actual bit rate

The resulting GateSpec Parameters as set by CMS would be:

Bucket Depth (b) = datagram size including IP/UDP/RTP-S header overhead = 202 bytes
Minimum Policed Unit (m) = Bucket Depth (b) = 202 bytes
Maximum Datagram Size (M) = Bucket Depth (b) = 202 bytes
Bucket Rate (r) = actual data rate, including IP/UDP/RTP-S header overhead = 10100 bytes per second
Peak Rate (p) = Bucket Rate (r) = 10100 bytes per second
Reserved Rate (R) = Bucket Rate (r) = 10100 bytes per second

Upstream DOCSIS parameters include overhead from the FC byte through the CRC.

DOCSIS base header (FC to HCS, no extended headers): 6 bytes
UGS Extended Header: 3 bytes
BPI+ Extended header: 5 bytes
Ethernet header: 14 bytes
CRC: 4 bytes
Total Upstream Overhead: 32 bytes per packet

DOCSIS Upstream Service Flow Parameters

Upstream Scheduling Type: UGS
Request/Transmission Policy (bitmask): bits 0-6, 8 set (101111111 binary)
Grant Size: 234 bytes
Grants per Interval (integer): 1
Grant Interval: 20000 microseconds
Tolerated Grant Jitter: 800 microseconds

The CMTS Authorization control procedure for the upstream parameters is conducted as follows:

To compare with GateSpec parameters, MAC-layer overhead must be subtracted from DOCSIS parameters.

GateSpec Bucket Depth (b) ≥ DOCSIS Unsolicited Grant Size – 32 bytes
202 bytes ≥ 234 bytes – 32 bytes = 202 bytes

GateSpec Bucket Rate (r) ≥ 1/DOCSIS Grant Interval × (DOCSIS Unsolicited Grant Size – 32)

10.1 kbyte/s ≥ 1/20 ms × (234 bytes – 32 bytes) = 50 packets per second × 202 bytes per packet = 10.1 kbyte/s.

Downstream DOCSIS parameters include overhead from the byte following the HCS through the CRC.

Ethernet header: 14 bytes
CRC: 4 bytes
Total Downstream Overhead: 18 bytes per packet

DOCSIS Downstream Service Flow Parameters

Maximum Traffic Burst (minimum value of 1522): 1522 bytes
Maximum Sustained Rate: 88000 bits per second
Assumed Minimum Reserved Rate Packet Size: 220 bytes
Minimum Reserved Rate: 88000 bits per second
Traffic priority: 5.

The CMTS Authorization control procedure for the downstream parameters is conducted as follows:

Again, this overhead must be subtracted from the DOCSIS parameters in order to perform the GateSpec comparison. The procedure is straightforward (subtraction) for the DOCSIS Assumed Minimum Reserved Rate Packet Size parameter. However, adjusting the Minimum Reserved Rate parameter is a bit more involved.

GateSpec Minimum Policed Unit (m) ≥ DOCSIS Assumed Minimum Reserved Rate Packet Size – 18 bytes

202 bytes ≥ 220 bytes – 18 bytes = 202 bytes

GateSpec Bucket Rate (r) ≥ (DOCSIS Minimum Reserved Rate / (8 × DOCSIS Assumed Minimum Reserved Rate Packet Size)) × (DOCSIS Assumed Minimum Reserved Rate Packet Size – 18 bytes)

10.1 kbyte/s ≥ (88 kbit/s / (8 × 220 bytes)) × (220 bytes – 18 byte/s) = 10.1 kbyte/s.

7.2.6 Authorization block encoding

The authorization block consists of a string of bytes. To allow for flexibility, the authorization block MUST be encoded using Type-Length-Value (TLV) fields. The TLV-tuple fields are unordered, and may be nested. The size of the value field (bytes) must be greater than zero; the sizes of the

type and length field are each one byte. Note that the length only includes the value field and not the entire TLV-tuple.

The format of the authorization block is as follows:

IPCablecom authorization block encoding

This field defines the parameters associated with the IPCablecom authorization block. Note that this field consists of nested sub-fields.

Type	Length	Value
1	n	"see sub-fields below"

Gate-id encoding

The value of this field specifies the gate-id handle used for authorization.

Type	Length	Value
[1].1	4	gate-id

Resource-id encoding

The value of this field specifies the resource-id handle used to uniquely identify the set of resources associated with a serviceflow.

Type	Length	Value
[1].2	4	resource-id

7.3 Use of J.112 MAC control service interface

The J.112 QoS parameters for the Service Flow derived from the SDP description are signalled to establish the Service Flow(s). In this clause, we describe how this can be done using the J.112 MAC control service interfaces (Annex E to Annex B/J.112).

At the level of J.112 MAC Control Service Interface primitives, the Embedded MTA signals for QoS resources as follows:

1) MAC_CREATE_SERVICE_FLOW.request:

   As described in B.E.3.2/J.112, the Embedded MTA can request that a Service Flow be added via this primitive. This primitive may also be used to define classifiers for the new Service Flow, as well as supply the Admitted and Active QoS Parameter Sets of the Service Flow. The success or failure of the primitive is indicated via the MAC_CREATE_SERVICE_FLOW.response primitive.

2) MAC_CHANGE_SERVICE_FLOW.request:

   The Embedded MTA can initiate a change in the Admitted and Active QoS Parameter Sets via this primitive. One possible scenario is the case of putting a caller on hold. The success or failure of the primitive is indicated via the MAC_CHANGE_SERVICE_FLOW.response primitive.

3) MAC_DELETE_SERVICE_FLOW.request :

   When the Embedded MTA no longer needs the Service Flow, it issues a MAC_DELETE_SERVICE_FLOW.request to the Embedded CM to zero the Active and Admitted QoS Parameter Sets of the Service Flow.

The parameters of these primitives match the parameters associated with the DSA, DSC, and DSD messages as given in Annex B to J.112.

7.3.1 Reservation establishment

The MTA initiates the reservation of QoS resources by use of the MAC_CREATE_SERVICE_FLOW.request primitive. The MTA MUST include the Gate-ID in the Authorization Block TLV. Upon reception of this message, the MAC layer of the CM invokes DSA signalling by sending a DSA_REQ to the CMTS. The CMTS MUST check the authorization based on the Gate-ID (contained in the Authorization Block TLV), and reject the request if the gate is invalid or the authorized resources are insufficient for the request. Upon receiving the DSA_RSP from the CMTS, the MAC service notifies the upper layer using the MAC_CREATE_SERVICE_FLOW.response message. This is illustrated in Figure 11.

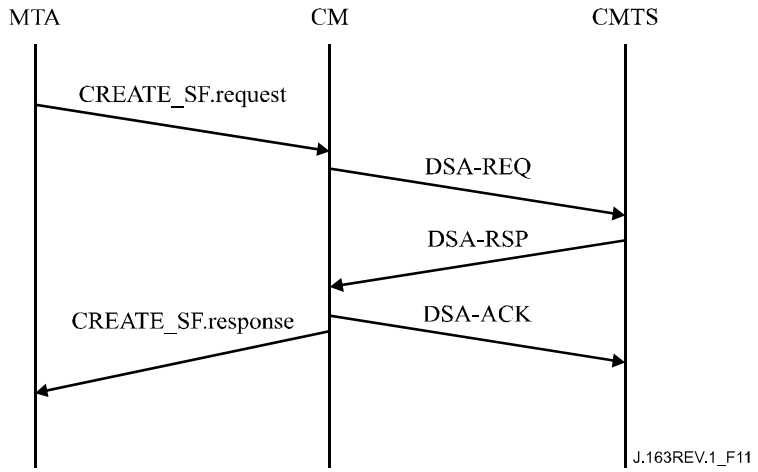


Figure 11/J.163 – Reservation establishment

7.3.2 Reservation change

The MTA initiates changes in QoS resources by use of the MAC_CHANGE_SERVICE_FLOW.request primitive. This is illustrated in Figure 12.

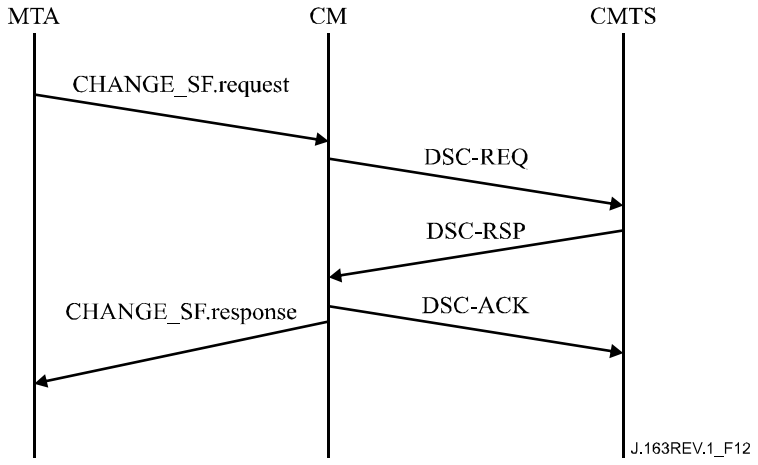


Figure 12/J.163 – Reservation change

Upon reception of this message, the MAC layer of the CM invokes DSC signalling. Upon receiving the DSC_RSP from the CMTS, the MAC service notifies the upper layer using the MAC_CHANGE_SERVICE_FLOW.response message.

7.3.3 Reservation deletion

The MTA initiates the deallocation of QoS reservation by use of the MAC_DELETE_SERVICE_FLOW.request primitive. Upon reception of this message, the MAC layer invokes DSD signalling. Upon receiving the DSD_RSP from the CMTS, the MAC service notifies the upper layer using the MAC_DELETE_SERVICE_FLOW.response message. This is illustrated in Figure 13.

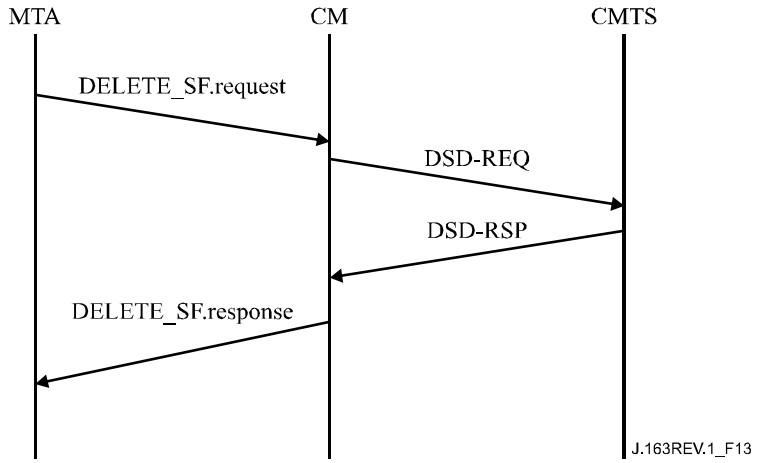


Figure 13/J.163 – Reservation deletion

8 Authorization interface description (pkt-q6)

This clause describes the interfaces between the CMTS and Gate Controller for purposes of authorizing the MTA to receive high Quality of Service. Signalling is required between the Gate Controller and CMTS to support gate management and IPCablecom QoS Admission Control Service. In addition, accurate subscriber billing requires the CMTS to indicate actual "committed" QoS resource usage on a per session basis. This clause describes the use of the COPS protocol to transport IPCablecom QoS defined messages between the Gate Controller and CMTS.

8.1 Gates: the Framework for QoS control

A IPCablecom Dynamic QoS "Gate" is a policy control entity implemented at the CMTS to control access to enhanced QoS Services of a J.112 network by a single IP flow. Gates are unidirectional, in that a single gate controls access to a flow in either the upstream or downstream direction. Gates enable the creation of J.112 Flow Classifiers, which in turn control the routing of packets to J.112 Flows.

While a Gate also has a N-tuple just like a Classifier, it is not identical to a Classifier. The CMTS MUST set up the Gate when a flow is authorized, until explicitly disabled to terminate the authorization for a flow. A J.112 Classifier MAY be set up and associated with a Gate. A Gate MAY exist before and after the Classifier it authorizes exists. A Gate MAY be considered to be associated with exactly zero, one, or two Classifiers.

A CMTS conforming to this Recommendation MUST NOT dynamically create a Classifier with a J.112 MAC message exchange unless it is authorized to do so by the existence of a Gate for that Classifier. An identifier, called the GateID is associated with Gates. The GateID, locally administered by the CMTS where the Gate exists, MAY be associated with one or more unidirectional Gates. For a point-to-point session, typically two unidirectional Gates exist, associated with a single GateID. In addition, J.112 Classifiers exist for each unidirectional flow that is established.

8.1.1 Classifier

A classifier is a six-tuple:

- Direction (Upstream/Downstream).
- Protocol.
- Source IP.
- Destination IP.
- Destination Port.
- Source Port.

If there is an upstream and an associated (part of the same session) downstream flow, then there MUST exist separate classifiers for the upstream flow and the downstream flow. The Classifier is updated by the RSVP message for the reservation performed for the upstream and downstream flows. The session data flow MUST match the classifier to receive the Quality of Service associated with the RSVP reservation.

8.1.2 Gate

A Gate is associated with a unidirectional flow, and comprises the following:

- Gate-ID.
- Prototype Classifier.
- Various flag bits described below.
- Authorized Envelope (Flow Spec).
- Reserved Envelope (Flow Spec).
- Resource-ID.

The GateID (described below) is a local 32-bit identifier that is allocated from the local space at the CMTS where the Gate resides. Up to two gates MAY share the same Gate-ID. Typically, a Gate-ID will identify a single upstream flow and a single downstream flow, and correspond to a single multimedia session. [This does not preclude bidirectional implementations, however.]

The Prototype Classifier consists of the same six elements as a Classifier, as described above. The Source IP is the IP address (as seen at the CMTS) of the originator of the flow. In the case of an upstream Gate on the J.112 channel, the Source IP is the IP address of the local MTA. For the downstream flow, the Source IP address is the IP address of the remote MTA. For selected parameters of a Gate's prototype classifier, a wild card is allowed. In Multimedia call signalling, the source UDP Port is not signalled, so its value is not considered to be part of a Gate's information.

The Source Port MAY be wildcarded, to support both IPCablecom Call Signalling Protocols (DCS and ITU-T Rec. J.162). If the Source Port is wildcarded, its value in the Gate parameters will be zero.

The Source IP address MAY be wildcarded, to support the J.162 Call Signalling Protocol. If the Source IP address is wildcarded, its value in the Gate parameters will be zero.

The Authorized and Reserved Envelopes are RSVP Flow Specs (both T-Spec and R-Spec) as described in previous clauses.

A reservation request for resources (as specified in the PATH message or equivalent J.112 MAC message) MUST be checked against what has been authorized for the Gate-ID associated with the direction for the resource request. The resources authorized are specified in the Authorized envelope. Also checked is the wildcard in the Gate for particular entries.

The Resource-ID is a local 32-bit identifier that is allocated from the local space at the CMTS where the Gate resides. Any number of gates MAY share a resource-ID, and therefore share a

common set of resources, with the restriction that only one of these gates in each direction has resources committed.

8.1.3 Gate identification

A GateID is a unique identifier that is locally allocated by the CMTS where the Gate resides. The GateID is a 32-bit identifier. A GateID MAY be associated with one or more Gates. In both the J.162 and DCS call signalling protocols, a Gate-ID is associated with each call leg, and consists of a single upstream gate and a single downstream gate.

A Gate-ID MUST be associated with the following information:

- One or two Gates, which MUST be one of the following combinations:
  - Single upstream gate.
  - Single downstream gate.
  - Single upstream gate and a single downstream gate [typically this would be a bidirectional implementation].
- Accounting and Billing information.
  - Address: Port of the Primary Record-Keeping-Server that should receive event records.
  - Address: Port of the Secondary Record-Keeping-Server, for use if the primary is unavailable.
  - Flag indicating whether the Event Messages are to be sent to the Record Keeping Server in real-time, or whether they are to be batched and sent at periodic intervals.
  - Billing-Correlation-ID, which will be passed to the Record-Keeping-Server with each event record.
  - Additional billing information, if supplied, which will be used to generate Call-Answer and Call-Disconnect event messages.
  - Omission of Event Generation information (i.e., the Event-Generation-Info object) implies that Event Message generation MUST NOT be performed for a Gate

The Gate-ID MUST be unique among all current gates allocated by the CMTS. The value of the 32-bit quantity SHOULD NOT be chosen from a set of small integers, since possession of the GateID value is a key element in the authentication of the COMMIT messages from the MTA. An algorithm that MAY be used to assign values of GateIDs is as follows: partition the 32-bit word into two parts, an index part, and a random part. The index part identifies the gate by indexing into a small table, while the random part provides some level of obscurity to the value. Regardless of the algorithm chosen, the CMTS SHOULD attempt to minimize the possibility of GateID ambiguities by ensuring that no GateID gets reused within three minutes of its prior closure or deletion. For the algorithm suggested previously this could be accomplished by simply incrementing the index part for each consecutively assigned GateID, wrapping around to zero when the maximum integer value of the index part is reached.

8.1.4 Gate transition diagram

Gates are considered to have the following states:

- Allocated – The initial state of a gate created at the request of the GC.
- Authorized – GC has authorized the flow with resource limits defined.
- Reserved – Resources have been reserved for the flow.
- Committed – Resources are being used.

The CMTS MUST support gate states and transitions as shown in Figure 14 and described in this clause. All gates assigned the same Gate-ID by the CMTS MUST transition together through the

states shown in Figure 14. This is true even when only one of the upstream/downstream flows is permitted to pass traffic.

In the interest of clarity, the gate transition diagram of Figure 14 does not completely describe all transitions that must be implemented, although all included transitions must be implemented as shown.

A gate is created in the CMTS by either a Gate-Alloc command or a Gate-Set command from the GC. In both cases, the CMTS allocates a locally unique identifier called a Gate-ID, which is returned to the GC. If the gate was created by a Gate-Set message, then the CMTS MUST mark the gate in state "Authorized" and MUST start Timer T1. If the gate was created by a Gate-Alloc message, then the CMTS MUST mark the gate in state "Allocated", start Timer T0, and MUST wait for a Gate-Set command, at which point the gate MUST be marked in state "Authorized". If the Timer T0 expires with the gate in state "Allocated" or Timer T1 expires with the gate in state "Authorized", then the CMTS MUST delete the gate. Timer T0 limits the amount of time the Gate-ID will remain valid without any specified gate parameters. Timer T1 limits the amount of time the authorization will remain valid.

A gate in the "Allocated" state MUST be deleted upon receipt of a Gate-Delete message. When this happens, the CMTS MUST respond with a Gate-Delete-Ack message and MUST stop timer T0. Similarly, a gate in the "Authorized" state MUST be deleted upon receipt of a Gate-Delete message. When this happens, the CMTS MUST respond with a Gate-Delete-Ack message and MUST stop timer T1.

A gate in the "Authorized" state is expecting the MTA to attempt to reserve resources. The MTA does this with either a RSVP-PATH message or via the MAC-layer Interface. On receipt of this reserve request, the CMTS MUST verify the request is within the limits established for the gate, and perform admission control procedures.

The CMTS MUST implement at least two admission control policies, one for normal voice communications and one for emergency communications. These two policies MUST have provisionable parameters that specify, at a minimum:

1) a maximum amount of resources that may be allocated non-exclusively to sessions of this type (which may be 100% of the capacity);
2) the amount of resources that may be allocated exclusively to sessions of this type (which may be 0% of the capacity); and
3) the maximum amount of resources that may be allocated to sessions of the two types.

The admission control policy MAY also specify whether a new session of that type may "borrow" from lower priority classes or should pre-empt an existing session of some other type to satisfy the admission control policy settings.

If the admission control procedures are successful, and only resource reservation was requested, the gate MUST be marked in the "Reserved" state. If admission control procedures are successful and single stage resource reservation and committal was requested, the gate MUST be marked in the "Committed" state and the CMTS MUST send a Gate-Open Message to the GC and stop timer T1.

If admission control procedures are not successful, the gate MUST remain in the "Authorized" state.

Note that the actual reservation made by the MTA may be for less than that authorized, e.g., reservation for upstream only when a pair of gates were established authorizing upstream and downstream flows.

In the "Reserved" state the gate is expecting the MTA to Commit to the resources, and thereby activate them. The Commit command from the MTA is either a unicast UDP message, or an equivalent request via the MAC-layer Interface. If the gate is still in the "Reserved" state and Timer T1 expires (i.e., the MTA does not issue the Commit command), the CMTS MUST release any resources reserved, and delete the gate. If a Gate-Delete message is received in the "Reserved"

state, the CMTS MUST respond with a Gate-Delete-Ack message, MUST release all resources associated with this gate, and MUST stop timer T1.

For the purpose of this state transition diagram, a "Commit" from the client is a message that commits the upstream flow. If the CMTS receives an asymmetric request such that traffic may pass on the downstream flow but not on the upstream flow, the CMTS MUST NOT move out of the "Reserved" state. If, on the other hand, the CMTS receives an asymmetric request such that traffic may pass on the upstream flow but not on the downstream flow, the CMTS MUST treat the request as a Commit and must transition its state in accordance with the description below.

If the T0 timer expires on the CMTS prior to receiving a Gate-Set command from the CMS, the CMTS MUST initiate a Gate-Close message using the "Timer T0 expiration; no Gate-Set received from CMS" as reason code, and delete the associated gate.

If the T1 timer expires on the CMTS prior to receiving a Commit command from the MTA, the CMTS MUST release any resources reserved with in association with the corresponding GateID, initiate a Gate-Close message using the "Timer T1 expiration; no COMMIT received from MTA" as reason code, and delete the associated gate(s).

If in the "Reserved" state the CMTS receives a Commit command from the client, the CMTS MUST mark the gate in the "Committed" state, stop timer T1, and initiate a Gate-Open message.

If the T7 timer expires and a service flow corresponding to the gate(s) referenced via the associated GateID has not been committed on the CMTS, the CMTS MUST initiate a Gate-Close message using the "Timer T7 expiration; Service Flow reservation timeout" as reason code, and delete the associated gate(s). Otherwise, the CMTS MUST set the reserved envelope equal to the committed envelope for flows corresponding to the gates referenced via the associated GateID.

If the T8 timer expires on the CMTS due to inactivity on the service flow, the CMTS MUST initiate a Gate-Close message using the "Timer T8 expiration; Service Flow inactivity in the upstream direction" as reason code, and delete the associated gate.

Once in the "Committed" state, the gate has reached a stable configuration. Resources have been at the local gates. Resources will continue to be committed until either the local MTA issues a Release command, the Active timer expires or the CMS issues a Gate-Delete command.

If, in the "Committed" state, the CMTS receives a Release command from the MTA, either in the form of a RSVP-PATH-TEAR message or via the MAC-layer interface, or from a failure of the client to refresh a reservation, or from internal J.112 mechanisms that detect a client failure, the CMTS MUST deactivate all resources committed for the MTA, release all resources reserved, initiate a Gate-Close message to the gate-coordination entity, and delete the gate.

If, in the "Committed" state, the CMTS receives a Gate-Delete message, the CMTS MUST deactivate all resources committed for the local client, release all resources reserved, and delete the gate. Additionally, the CMTS must respond with a Gate-Delete-Ack message.

While in the "Committed" state, the CMTS MUST allow the MTA to initiate changes in the resource reservation or commitment, within the limits of the authorization and local admission control.

8.1.5 Gate coordination

The Gate Coordination messages in the COPS Gate Control interface, Gate-Open and Gate-Close, provide an unsolicited feedback mechanism from the CMTS to the CMS in order to maintain state-synchronization between these elements. This is particularly useful in the case of a pre-mature MTA-initiated reservation or committal request which is not stimulated by the CMS or in the event that an MTA fails, initiating resource recovery at the CMTS. In both of these potential scenarios, the internal state maintained within the CMS will be updated to reflect the state change at the CMTS and the CMS will be able to take appropriate action based in this information.

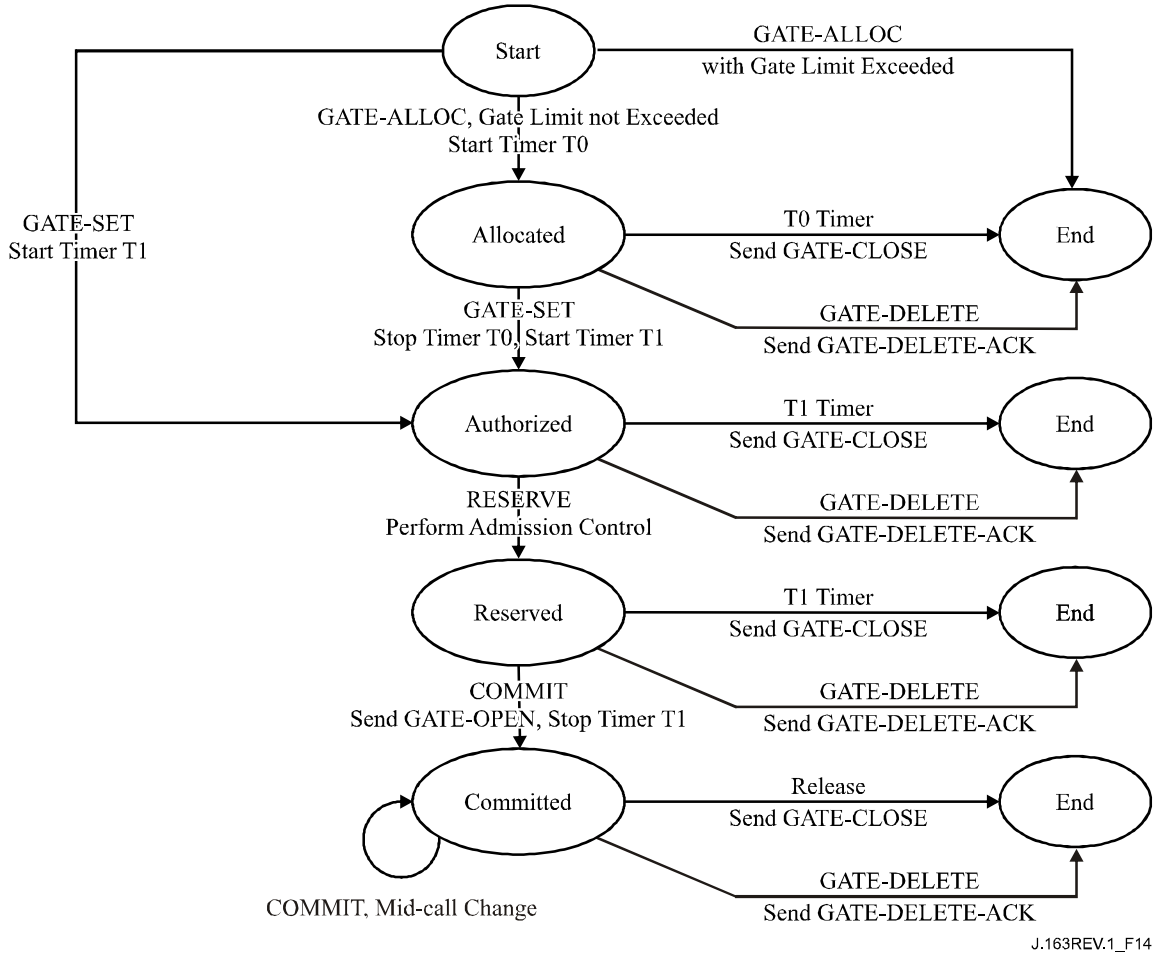


Figure 14/J.163 – Gate state transition diagram

8.2 COPS profile for IPCablecom

IPCablecom QoS Admission Control is the act of managing QoS resource allocation based on administrative policies and available resource. IPCablecom QoS Admission Control Service uses a client/server architecture. The high-level operational modules are depicted in Figure 15. The administrative policies are stored as policy database and controlled by the COPS Server. While a typical Intserv implementation of COPS has the server determine available resources, a Diffserv implementation pushes the policy into the client so that the client can make admission control decisions.

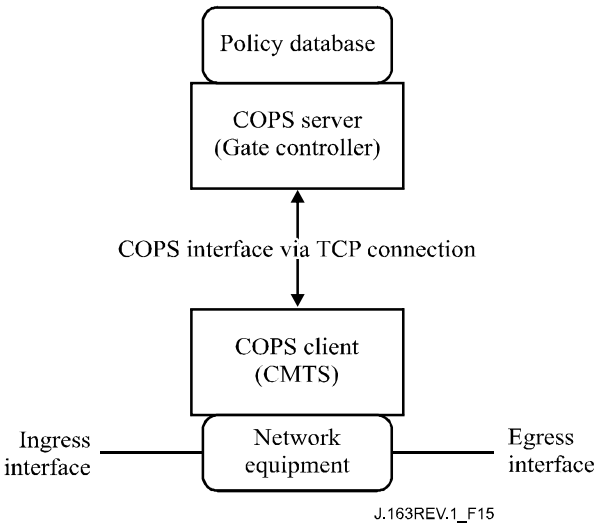


Figure 15/J.163 – QoS Admission Control layout

The QoS Admission Control decisions made by the COPS Server MUST be passed to the COPS Client using COPS. The COPS Client MAY make QoS Admission Control requests to the COPS Server based on network events triggered by either the QoS signalling protocol, or via data flow detection mechanisms. The network event can also be the need of QoS bandwidth management, e.g., a new QoS capable interface becomes operational.

QoS policy decisions made by the COPS Server MAY be pushed to the COPS Client based on an external, out-of-band, QoS service request, e.g., request from the terminating CMTS or a Gate Controller. These policy decisions MAY be stored by the COPS client in a local policy decision point and the CMTS may access that decision information to make admission control decisions on incoming session requests received at the CMTS.

The COPS Client-COPS Server interaction support for QoS Admission Control is provided by IETF's COPS protocol. The COPS protocol includes the following operations:

- Client-Open (OPN)/Client-Accept (CAT)/Client-Close (CC): The COPS Client sends an OPN message to initiate a connection with the COPS Server, and the Server responds with a CAT message to accept the connection. The server sends a CC message to terminate the connection with the Client.
- Request (REQ): The COPS Client sends a REQ message to the server to request admission control decision information or device configuration information. The REQ message may contain client-specific information that the server uses, together with data in the session admission policy database, to make policy-based decisions.
- Decision (DEC): The server responds to REQs by sending a DEC back to the client that initiated the original request. DEC messages may be sent immediately in response to a REQ (i.e., a solicited DEC) or at any time after to change/update a previous decision (i.e., an unsolicited DEC).
- Report State (RPT): The COPS Client sends a RPT message to the COPS Server indicating changes to the request state in the COPS Client. The COPS Client sends this to inform the COPS Server the actual resource reserved after the COPS Server has granted admission. The COPS Client can also use Report to periodically inform the COPS Server the current state of the COPS Client.
- Delete Request State (DEL): The COPS Client sends a DEL message to the COPS Server for request state cleanup. This can be the result of QoS resource release by the COPS Client.

- Keep Alive (KA): Sent by both the COPS Client and COPS Server for communication failure detection.
- Synchronize State Request (SSR)/Synchronize State Complete (SSC): SSR is sent by the COPS Server requesting current COPS Client state information. The client re-issues request queries to the server to perform the synchronization, and then the client sends a SSC message to indicate synchronization is complete. Because the GC is stateless, the SSR/SSC operations are of no importance in IPCablecom and are not used by the CMTS or GC.

Within the IPCablecom architecture, the Gate Controller is a COPS Policy Decision Point (i.e., PDP) entity and the CMTS is the COPS Policy Enforcement Point (i.e., PEP) entity.

The details of the COPS protocol are provided in RFC 2748. This RFC provides a description of the base COPS protocol, independent of client type. Additional drafts provide information for using COPS for Integrated Services with RSVP and for Differentiated Services (i.e., provisioning clients). A more detailed overview of the COPS protocol is provided as information in Appendix X.

8.3 Gate Control protocol message formats

Protocol messages for Gate Control are transported within the COPS protocol messages. COPS utilizes a TCP connection established between the CMTS and the Gate Controller, and will use the mechanisms specified in standards under development to secure the communication path.

8.3.1 COPS common message format

Each COPS message consists of the COPS header followed by a number of typed objects. The GC and CMTS MUST support COPS messaging as defined below (see Figure 16):

<table>
<tr><td colspan="2">0</td><td>1</td><td>2</td><td>3</td></tr>
<tr><td>Version</td><td>Flags</td><td>Op-Code</td><td colspan="2">Client-type</td></tr>
<tr><td colspan="4">Message length</td></tr>
</table>

Figure 16/J.163 – Common COPS message header

Version is a 4-bit field giving the current COPS version number. This MUST be set to 1.

Flags is a 4-bit field. 0x1 is the solicited message flag. When a COPS message is sent in response to another message (e.g., a solicited decision sent in response to a request) this flag MUST be set to 1. In other cases (e.g., an unsolicited decision) the flag MUST NOT be set (value = 0). All other flags MUST be set to zero.

Op-code is a 1-byte field that gives the COPS operation to be performed. COPS operations used in this IPCablecom specification are:

1 = Request (REQ)

2 = Decision (DEC)

3 = Report-State (RPT)

6 = Client-Open (OPN)

7 = Client-Accept (CAT)

9 = Keep-Alive (KA)

Client type is a 16-bit identifier. For IPCablecom use the Client type MUST be set to IPCablecom client (0x8008). For Keep-Alive messages (Op-code = 9) the client-type MUST be set to zero, as the KA is used for connection verification rather than per client session verification.

Message length is a 32-bit value giving the size of the message in octets. Messages MUST be aligned on 4-byte boundaries, so the length MUST be a multiple of four.

Following the COPS common header are a variable number of objects. All the objects follow the same object format; each object consists of one or more 32-bit words with a four-octet header, using the following format (see Figure 17):

0	1	2	3
Length		C-Num	C-type
(Object contents)			

Figure 17/J.163 – Common COPS object format

The length is a two-octet value that MUST give the number of octets (including the header) that compose the object. If the length in octets is not a multiple of four, padding MUST be added to the end of the object so that it is aligned to the next 32-bit boundary. On the receiving side, a subsequent object boundary MUST be found by rounding up the previous stated object length to the next 32-bit boundary.

C-Num identifies the class of information contained in the object, and the C-Type identifies the subtype or version of the information contained in the object. Standard COPS objects (as defined in RFC 2748) used in this Recommendation, and their values of C-Num, are:

1 = Handle

6 = Decision

8 = Error

9 = Client Specific Info

10 = Keep-Alive-Timer

11 = PEP Identification

8.3.2 Additional COPS Objects for Gate Control

As with the COPS-PR and COPS-RSVP client types, the IPCablecom client type defines a number of object formats. These objects MUST be placed inside a Decision object, C-Num = 6, C-Type = 4 (Client specific Decision Data) when carried from GC to CMTS in a decision message. They MUST also be placed in a ClientSI object, C-Num = 9, C-Type = 1 (Signalled Client SI) when carried from CMTS to GC in a Report message. They are encoded similarly to the client-specific objects for COPS-PR; detailed encodings appear below. As in COPS-PR, these objects are numbered using a client-specific number space, which is independent of the top-level COPS object number space. For this reason, the object numbers and types are given as S-Num and S-Type respectively.

Additional COPS objects defined for use by IPCablecom are as follows:

8.3.2.1 Transaction-ID

The Transaction-ID contains a token that is used by the GC to match responses from the CMTS to the previous requests, and the command type that identifies the action to be taken or response.

Length = 8	S-Num = 1	S-Type = 1
Transaction Identifier	Gate Command Type	

Transaction Identifier is a 16-bit quantity that MAY be used by the GC to match responses to commands.

Gate Command Type MUST be one of the following:

GATE-ALLOC	1
GATE-ALLOC-ACK	2
GATE-ALLOC-ERR	3
GATE-SET	4
GATE-SET-ACK	5
GATE-SET-ERR	6
GATE-INFO	7
GATE-INFO-ACK	8
GATE-INFO-ERR	9
GATE-DELETE	10
GATE-DELETE-ACK	11
GATE-DELETE-ERR	12
Gate-Open	13
Gate-Close	14

8.3.2.2 Subscriber-ID

The Subscriber-ID identifies the subscriber for this service request. Its main use is to prevent various denial-of-service attacks.

Length = 8	S-Num = 2	S-Type = 1
IPv4 address (32 bits)		

or:

Length = 20	S-Num = 2	S-Type = 2
IPv6 address (128 bits)		

8.3.2.3 Gate-ID

This object identifies the gate or set of gates referenced in the command message, or assigned by the CMTS for a response message.

Length = 8	S-Num = 3	S-Type = 1
Gate-ID (32 bits)		

8.3.2.4 Activity-Count

When used in a GATE-ALLOC message, this object specifies the maximum number of gates that can be simultaneously allocated to the indicated subscriber-ID. This object returns, in a GATE-SET-ACK or GATE-ALLOC-ACK message, the number of gates assigned to a single subscriber. It is useful in preventing denial-of-service attacks.

Length = 8	S-Num = 4	S-Type = 1
Count (32 bits)		

8.3.2.5 Gate-spec

Length = 60		S-Num = 5	S-Type = 1	
Direction	Protocol ID	Flags, defined below	Session Class	
Source IP Address (32 bits)				
Destination IP Address (32 bits)				
Source Port (16 bits)		Destination Port (16 bits)		
DiffServ Code Point (DSCP)				
Timer T1 value		Reserved		
Timer T7 value		Timer-T8 value		
Token Bucket Rate [r] (32-bit IEEE floating point number)				Flow spec alt #1
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)				
Rate [R] (32-bit IEEE floating point number)				
Slack Term [S] (32-bit integer)				
Additional sets of r, b, p, m, M, R, and S values, as needed, to describe the authorization.				Flow spec alt #2, etc.

Direction is either 0 for a downstream gate, or 1 for an upstream gate.

Protocol-ID is the value to match in the IP header, or zero for no match.

Auto-Commit and Commit-Not-Allowed functionality which was formerly signalled through the flags field has been deprecated. As a result, bits one and two are reserved. All bits MUST be zero.

Session class identifies the proper admission control policy or parameters to be applied for this gate. Permissible values are:

0x00 Unspecified.

0x01 Normal priority VoIP session.

0x02 High priority VoIP session (e.g., E911).

All other values are currently reserved.

Source IP Address and Destination IP Address are a pair of 32-bit IPv4 addresses, or zero for no match (i.e., a wildcard specification that will match any request from the MTA).

Source Port and Destination Port are a pair of 16-bit values, or zero for no match.

The values of r, b, p, m, M, R, and S, are as described in 6.2. Instead of the RSVP RFC defined slack-term the value S would represent in microseconds the minimum allowed grant jitter in the upstream direction that can be admitted, and the minimum allowed delay in the downstream direction that can be admitted.

Other clauses provide normative requirements which represent constraints on the authorization envelope which is defined by these parameters. Specifically, the multiple codec discussion in 5.6.10 defines an upper bound on the authorization envelope, while clause 8.5 provides a set of minimal requirements for these parameters. It is strongly recommended that CMS implementations constrain authorization parameters as much as possible as these constructs are fundamental in defining and enforcing service provider bandwidth management policies.

The DS field is defined by the following structure:

0	1	2	3	4	5	6	7
Differentiated Services Code Point (DSCP)						Not used	Not used

For backward compatibility with current system implementations and use of the IP Precedence as defined in IETF RFC 2474 and IETF RFC 791, the appropriate bits of the IPv4 TOS byte shown below MAY be inserted in the DS field. The IP TOS field (bits 3-6) is not supported in Diffserv networks.

0	1	2	3	4	5	6	7
IP Precedence			IPv4 IP TOS				Not used

Timer T1 is given in seconds, and used in the Gate Transition Diagram described in 8.1.4. If multiple Gate-Spec objects appear in a single COPS message, the values of T1 MUST be identical in all Gate-Spec occurrences. If the T1 values differ in the upstream and downstream GateSpec objects, then the CMTS MUST use the T1 value specified in the upstream GateSpec to manage the pair of Gates.

Timers T7 and T8 are values in seconds and used to control the DOCSIS Timeout for Admitted QoS Parameters and Timeout for Active QoS Parameters respectively.

8.3.2.6 Remote-Gate-Info

This object is no longer valid. S-Num 6 is reserved to prevent misinterpretation.

Length 36	S-Num = 6	S-Type = 1
CMTS IP Address (32 bits)		
CMTS Port (16 bits)	Flags, defined below	
Remote Gate-ID		
Algorithm	Reserved	
Security Key (16 bytes)		

8.3.2.7 Event-Generation-Info

The object contains all the information necessary to support the event messages as specified and required in ITU-T Rec. J.164.

Length = 44	S-Num = 7	S-Type = 1
Primary-Record-Keeping-Server-IP-Address (32 bits)		
Primary-Record-Keeping-Server-Port	Flags, see below	Reserved
Secondary-Record-Keeping-Server-IP-Address (32 bits)		
Secondary-Record-Keeping-Server-Port	Reserved	
Billing-Correlation-ID (24 bytes)		

Primary-Record-Keeping-Server-IP-Address is the address of the record keeper to whom event records are sent.

Primary-Record-Keeping-Server-Port is the port number for event records sent.

Flag values are as follows:

0x01 Batch processing indicator. If set, the CMTS MUST accumulate event records as part of a batch file and send to Record Keeping Server at periodic intervals. If clear, the CMTS MUST send the event records to the Record Keeping Server in real-time.

The rest are reserved and MUST be zero.

Secondary-Record-Keeping-Server-IP-Address is the address of the secondary record keeper to whom records are sent if the primary record keeping server is unavailable.

Secondary-Record-Keeping-Server-Port is the port number for event records sent.

Billing-Correlation-ID is the identifier assigned by the CMS for all records related to this session.

8.3.2.8 Media-Connection-Event-Info

This object is no longer required. S-Num 8 is reserved to prevent misinterpretation.

8.3.2.9 IPCablecom-Reason

This object contains the reason why the gate is being deleted.

Length = 8	S-Num = 13	S-Type = 1
Reason-code	Reason Sub-code	

The Reason-code values defined in this Recommendation are:

0: Gate-Delete Operation

1: Gate-Close Operation

The Reason Sub-codes are defined as:

Gate-Delete Operation:

0 = Normal operation

1 = Local gate-coordination not completed

2 = Remote gate-coordination not completed

3 = Authorization revoked

4 = Unexpected Gate-Open

5 = Local Gate-Close failure

127 = Other, unspecified error

Gate-Close Operation:

0 = Client initiated release (normal operation)

1 = Reservation reassignment (e.g., for priority session)

2 = Lack of reservation maintenance (e.g., RSVP refreshes)

3 = Lack of DOCSIS MAC-layer responses (e.g., station maintenance)

4 = Timer T0 expiration; no Gate-Set received from CMS

5 = Timer T1 expiration; no COMMIT received from MTA

6 = Timer T7 expiration; Service Flow reservation timeout

7 = Timer T8 expiration; Service Flow inactivity in the upstream direction

127 = Other, unspecified error

8.3.2.10 IPCablecom-Error

A client-specific error object is defined as follows:

Length = 8	S-Num = 9	S-Type = 1
Error-code	Error Sub-code	

The Error-code values defined in this Recommendation are:

1 = No gates currently available.

2 = Unknown Gate-ID.

3 = Illegal Session Class value.

4 = Subscriber exceeded gate limit.

6 = Missing Required Object

7 = Invalid Object

127 = Other, unspecified error.

The Error Sub-code field is used to provide further information about the error. In the case of error codes 6 through 7, this 16-bit field contains as two 8-bit values the S-Num and S-type of the object that is missing or in error. The order of the S-Num and S-Type values within the Error Sub-code MUST be the same as in the original message. In cases where multiple valid alternatives exist for the S-Type of a missing object, this portion of the Error Sub-code should be set to 0.

8.3.2.11 Electronic-Surveillance-Parameters

Length = 24	S-Num = 10	S-Type = 1
DF-IP-Address-for-CDC (32 bits)		
DF-Port-for-CDC (16 bits)	Flags, defined below	
DF-IP-Address-for-CCC (32 bits)		
DF-Port-for-CCC (16 bits)	Reserved	
CCCID (32 bits)		

DF-IP-Address-for-CDC is the address of the Electronic Surveillance Delivery Function to whom the duplicated event messages are to be sent.

DF-Port-for-CDC is the port number for the duplicated event messages.

Flags are defined as follows:

0x0001 DUP-EVENT. If set, CMTS MUST send a duplicate copy of all event messages related to this gate to the DF-IP-Address-for-CDC.

0x0002 DUP-CONTENT. If set, CMTS MUST send a duplicate copy of all packets matching the classifier(s) for this gate to the DF-IP-Address-for-CCC.

The rest are reserved and MUST be zero.

DF-IP-Address-for-CCC is the address of the Electronic Surveillance Delivery Function to whom the duplicated call content packets are to be sent.

DF-Port-for-CCC is the port number for the duplicated call content.

CCCID is the identifier for duplicated call content packets.

8.3.2.12 Session-Description-Parameters

This object is no longer used. S-Num 11 is reserved to prevent misinterpretation.

Length =	S-Num = 11	S-Type = 1

8.3.3 Definition of Gate Control Messages

Messages that perform gate control between the GC and CMTS MUST be defined and formatted as follows. Note that messages from GC to CMTS are COPS Decision messages, and messages from CMTS to GC are COPS Report messages.

<Gate-Control-Cmd> := <COPS-Common-Header> <Handle>
<Context> <Decision Flags>
<ClientSI-Data>

<ClientSI-Data> := <Gate-Alloc> | <Gate-Set> | <Gate-Info>> |
<Gate-Delete>

<Gate-Control-Response> ::= <COPS-Common-Header> <Handle>
<Report-Type> <ClientSI-Object>

<ClientSI-Object> ::= <Gate-Alloc-Ack> | <Gate-Alloc-Err> |
<Gate-Set-Ack> | <Gate-Set-Err> |
<Gate-Info-Ack> | <Gate-Info-Err> |
<Gate-Delete-Ack> | <Gate-Delete-Err>

<Gate-Alloc> := <Decision-Header> <Transaction-ID>
<Subscriber-ID>[<Activity-Count>]

<Gate-Alloc-Ack> ::= <ClientSI-Header> <Transaction-ID> <Subscriber-ID>
<Gate-ID> <Activity-Count>>

<Gate-Alloc-Err> ::= <ClientSI-Header> <Transaction-ID> <Subscriber-ID>
<IPCablecom-Error>

<Gate-Set> ::= <Decision-Header> <Transaction-ID> <Subscriber-ID>
[<Activity-Count>] [<Gate-ID>]
[<Event-Generation-Info>]
[<Electronic-Surveillance-Parameters>]
<Gate-Spec> [<Gate-Spec>]

<Gate-Set-Ack> ::= <ClientSI-Header> <Transaction-ID> <Subscriber-ID>
<Gate-ID> <Activity-Count>

<Gate-Set-Err> ::= <ClientSI-Header> <Transaction-ID> <Subscriber-ID>
<IPCablecom-Error>

<Gate-Info> := <Decision-Header> <Transaction-ID> <Gate-ID>

<Gate-Info-Ack> := <ClientSI-Header> <Transaction-ID> <Subscriber-ID>
<Gate-ID>
[<Event-Generation-Info>][<Electronic-Surveillance-Parameters>]
[<Gate-Spec>] [<Gate-Spec>]

<Gate-Info-Err> ::= <ClientSI-Header> <Transaction-ID> <Gate-ID>
<IPCablecom-Err>

<Gate-Delete> ::= <Decision-Header> <Transaction-ID> <Gate-ID><IPCablecom
reason>

<Gate-Delete-Ack> := <ClientSI-Header> <Transaction-ID> <Gate-ID>

<Gate-Delete-Err> := <ClientSI-Header> <Transaction-ID> <Gate-ID>
<IPCablecom-Err>

<Gate-Open> :: = <ClientSI-Header> <TransactionID> <GateID>

<Gate-Close> :: = <ClientSI-Header> <TransactionID> <GateID>
<IPCablecom-Reason>

The Context object (C-NUM = 2, C-TYPE = 1) in the COPS Decision message has the R-Type (Request Type Flag) value set to 0x08 (Configuration Request) and the M-Type set to zero. The Command-Code field in the mandatory Decision-Flags object (C-NUM = 6, C-TYPE = 1) is set to 1 (Install Configuration). Other values should cause the CMTS to generate a Report message indicating failure. The Report-Type object (C-NUM = 12, C-TYPE = 1) included in the COPS Report message has the Report-Type field set to 1 (Success) or 2 (Failure) depending on the outcome of the gate control command. All Report messages carrying the gate control response should have the solicited message flag bit set in the COPS header. All decision (DEC) messages, except the first one, should have the solicited message flag set to false in the COPS header. The first decision message sent from the CMS to CMTS should have the solicited flag set to true. The values of this flag are set to comply with the COPS specification. They should not affect the operation of the gate control protocol.

If an object, that is received in a gate control message, contains an S-Num or S-Type that is not recognized that object MUST be ignored. The presence of such an object within a gate control message MUST not be treated as an error provided that after such parameter is dropped, all required objects are present in the message

8.4 Gate control protocol operation

8.4.1 Initialization sequence

When the CMTS (i.e., COPS PEP) boots, it MUST listen for incoming COPS connections on TCP port 2126 (assigned by IANA). Any Gate Controller with a need to contact the CMTS MUST establish a TCP connection to the CMTS on that port. It is expected that multiple Gate Controllers will establish COPS connections with a single CMTS. When the TCP connection between the CMTS and GC is established, the CMTS sends information about itself to the GC in the form of a CLIENT-OPEN message. This information includes the provisioned CMTS-ID in the PEP Identification (PEPID) object. The CMTS SHOULD omit the Last PDP Address (LastPDPAddr) object from the CLIENT-OPEN message.

In response, the Gate Controller sends a CLIENT-ACCEPT message. This message includes the Keep-Alive-Timer object, which tells the CMTS the maximum interval between Keep-Alive messages.

The CMTS then sends a REQUEST message, including the Handle and Context objects. The Context object (C-NUM = 2, C-TYPE = 1) MAY have the R-Type (Request Type Flag) value set to 0x08 (Configuration Request) and M-Type set to zero. The Handle object contains a number, that is chosen by the CMTS. The only requirement imposed on this number is that an CMTS MUST NOT use the same number for two different REQUESTs on a single COPS connection; in the IPCablecom environment the handle has no other protocol significance. This completes the initialization sequence, which is shown in Figure 18.

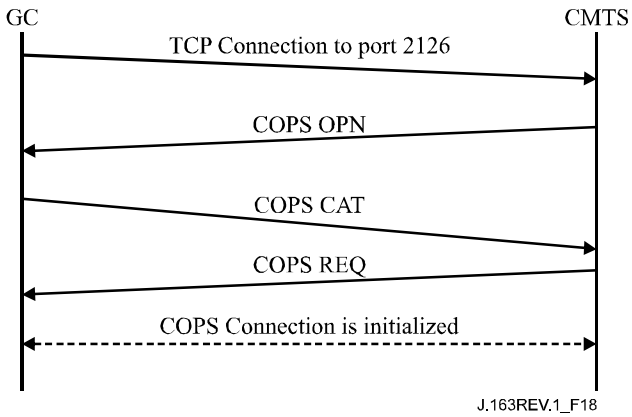


Figure 18/J.163 – COPS Connection establishment

Periodically the CMTS MUST send a COPS KEEP-ALIVE (KA) message to the GC. Upon receipt of the COPS KA message, the GC MUST echo a COPS KA message back to the CMTS. This transaction is shown in Figure 19 and is fully documented in IETF RFC 2748. This MUST be done at least as often as specified in the Keep-Alive-Timer object returned in the CLIENT-ACCEPT message. The KEEP-ALIVE message is sent with Client-Type set to zero.

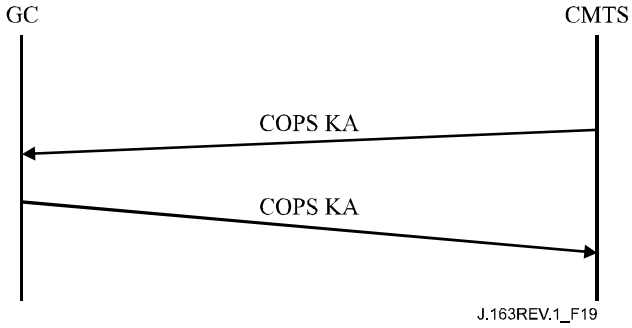


Figure 19/J.163 – COPS Keep-alive exchange

8.4.2 Operation sequence

The protocol between the Gate Controller and CMTS is for purposes of resource control and resource allocation policy. The Gate Controller implements all the allocation policies, and uses that information to manage the set of gates implemented in the CMTS. The Gate Controller initializes the gates with specific source, destination, and bandwidth restrictions; and once initialized, the MTA is able to request resource allocations within the limits imposed by the Gate Controller.

Messages initiated by the Gate Controller include GATE-ALLOC, GATE-SET, GATE-INFO, GATE-DELETE and messages initiated by the CMTS include Gate-Open and Gate-Close. The procedures for these messages are described in the following clauses.

The GC-initiated messages are sent using client specific objects within the decision object of COPS DECISION messages. The responses to the GC initiated messages are sent as a REPORT-STATE message with client specific objects in the ClientSI object by the CMTS. For the ACK messages the COPS Report-Type value MUST be 1 and for the ERR messages the Report-Type MUST be 2. Gate-Open and Gate-Close messages MUST be sent as an unsolicited REPORT-STATE message with transaction ID of zero, with client specific objects in the ClientSI object, using the Report Type 3, to the CMS through the TCP connection that originally constructed the gate. If that TCP connection is no longer valid, then the CMTS MUST silently drop the GC messages.

The DECISION messages and REPORT-STATE messages MUST contain the same handle as was used in the initial REQUEST sent by the CMTS when the COPS connection was initiated.

GATE-ALLOC validates the number of simultaneous sessions allowed to be set up from the originating MTA, and allocates a Gate-ID to be used for all future messages regarding this gate or set of gates.

GATE-SET initializes and modifies all the policy and traffic parameters for the gate or set of gates, and sets the billing and gate coordination information.

GATE-INFO is a mechanism by which the Gate Controller can find out all the current state and parameter settings of an existing gate or set of gates.

The CMTS MUST periodically send a Keep Alive (KA) message to the GC to facilitate the detection of TCP connection failures. The Gate Controller keeps track of when KAs are received. If the Gate Controller has not received a KA from the CMTS in the time specified by IETF RFC 2748 or the Gate Controller has not received an error indication from the TCP connection, then the Gate Controller MUST tear down the TCP connection and attempt to re-establish the TCP connection before the next time it is requested to allocate a gate from that CMTS.

GATE-DELETE allows a Gate Controller to delete a recently allocated gate under certain (see below) circumstances.

Gate-Open allows the CMTS to inform the Gate-Controller that the gate resources have been committed. The Gate-Open message, along with the Gate-Close message described below, provide a feedback path from CMTS to CMS in order to allow for accurate call-state management at the CMS element.

8.4.3 Procedures for allocating a new gate

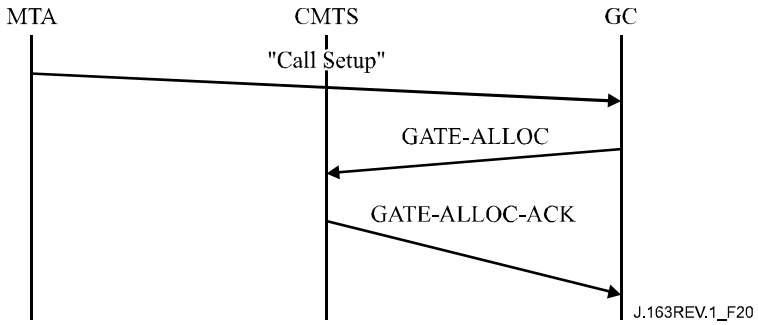
A GATE-ALLOC message is sent by the Gate Controller to the CMTS at the time the "Call_Set-up" message is sent from the originating MTA (e.g., "Invite(stage1)" message when using DCS), as shown in Figure 19.

The use of GATE-ALLOC ensures that not too many sessions are being simultaneously requested from a given MTA. This mechanism may be used to control a denial-of-service attack from the MTA. The CMTS, in its response to the GATE-ALLOC message, compares the number of currently allocated gates for the indicated subscriber-ID against the Count field of the Activity-Count object in the GATE-ALLOC message. If the current number of gates is greater than or equal to the Count field in the GATE-ALLOC, then the CMTS MUST return a GATE-ALLOC-ERR message. If the current number of gates is greater than the Count field in the GATE-ALLOC, then it is likely that the subscriber has been re-provisioned to have a lower gate limit than before. In this case, the subscriber's current sessions are not affected but any new sessions by that subscriber will be rejected by the CMTS until the subscriber's session count goes below the value specified in the Count field.

The determination of the actual value to be contained in the Count field is an operational issue. It should be set sufficiently high (per MTA) that no possible legitimate calling scenario is adversely affected, while being sufficiently low as to prevent a viable denial-of-service attack to be mounted.

If the Activity-Count object is not present, the CMTS does not perform the gate limit check. A GC seeking to reduce call set-up time MAY decide to perform the gate limit check upon receipt of the GATE-ALLOC-ACK instead of having the CMTS perform the check so that the GC can do the GATE-ALLOC and subscriber policy lookup operations in parallel. When the results of both operations are available, the GC can do the gate limit check. If the check fails, the GC SHOULD send a GATE-DELETE message to the CMTS to delete the gate that was incorrectly allocated (see 8.4.8). The GC MAY include the Activity-Count object in subsequent GATE-ALLOCs for that subscriber once the policy has been cached.

The following diagram (see Figure 20) is an example of the GATE-ALLOC signalling:



NOTE – As an example, the "Call Setup" message in this context refers to the "Invite w/o ring" when using DCS.

Figure 20/J.163 – Sample signalling of GATE-ALLOC

The CMTS MUST respond to a GATE-ALLOC message with either a GATE-ALLOC-ACK (indicating success) or a GATE-ALLOC-ERR (indicating failure). The Transaction-ID in the response MUST match the transaction ID in the request.

Errors in allocating gates are reported by a GATE-ALLOC-ERR response. The IPCablecomError object contains one of the following Error-Codes:

1 = No gates currently available.

4 = Subscriber exceeded gate limit.

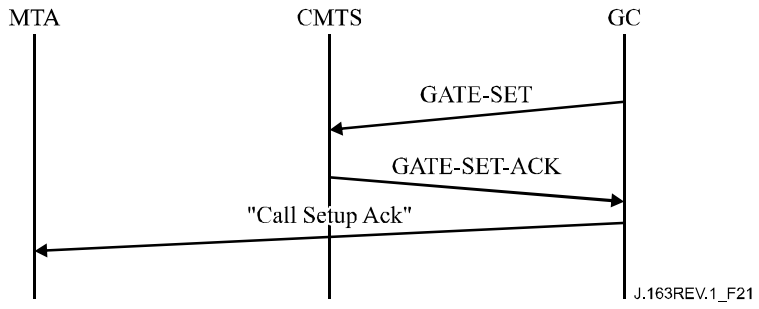
6 = Missing Required Object

7 = Invalid Object

127 = Other, unspecified error.

8.4.4 Procedures for authorizing resources through a gate

The GATE-SET message is sent by the Gate Controller to the CMTS to initialize or modify the operational parameters of the gate(s). Figure 21 is an example of the GATE-SET signalling.



NOTE – As an example, the "Call Setup Ack" message in this context refers to the "200 OK" message returned from the "Invite w/o ring" when using DCS.

Figure 21/J.163 – Sample signalling of GATE-SET

If a Gate-ID Object is present in the GATE-SET message, then the request is to modify an existing gate. If the Gate-ID Object is missing from the GATE-SET message, then it is a request to allocate a new gate, and the Activity-Count Object MAY be present so that the CMTS can determine if the subscriber has exceeded the maximum number of simultaneous gates (see 8.4.3).

The GATE-SET message MUST contain exactly one or two Gate-Spec objects, describing zero or one upstream gates, and zero or one downstream gates.

The CMTS MUST respond to a GATE-SET message with either a GATE-SET-ACK (indicating success) or a GATE-SET-ERR (indicating failure). The Transaction-ID in the response MUST match the transaction ID in the request.

Errors in allocating or authorizing gates are reported by a GATE-SET-ERR response. The IPCablecom-Error object contains one of the following Error-Codes:

1 = No gates currently available.

2 = Illegal Gate-ID.

3 = Illegal Session Class value.

4 = Subscriber exceeded gate limit.

5 = The gate is already set.

6 = Missing Required Object

7 = Invalid Object

127 = Other, unspecified error.

In handling a reservation request from an MTA, the CMTS MUST determine the proper gate by use of the RSVP Gate-ID object, or by the use of the Authorization Block TLV. The CMTS MUST verify the reservation request is within the authorized limits specified for the gate.

The CMTS then updates the reservation request based on gate parameters. If the auto-commit flag is set, then the CMTS MUST take appropriate action on the J.112 MAC-layer to commit the resources immediately. The CMTS MUST use the Diffserv Code Point or TOS value from the Gate-Spec object to overwrite the IP Type of Service octet before forwarding packets.

The CMTS MUST perform an admission control function, based on provisioned policy parameters and the Session Class value of the gate.

Note that a GATE-SET message can be used to allocate (and set) a gate instead of the GATE-ALLOC message. In such situations, it is possible that the port number being used by the remote gate for receiving gate coordination messages is not available to the gate controller. If that is the case, the CMTS-port in the Remote-Gate-Info object (carried in the GATE-SET message) is set to zero. This causes the CMTS to ignore the gate coordination port number. However, when the gate controller (later) learns about the port number being used by the remote gate, it must send another GATE-SET message (with the port number in the Remote-Gate-Info object) to inform the CMTS about this port.

The intention of a Gate-Set is that the most recent parameter values would be used for admission control when moving a gate from state Authorized to state Reserved. Once resources have been reserved, the MTA is guaranteed that any commit operation within the reserved envelope would succeed. After this time (i.e., the gate state is Reserved, or Committed), the gate MUST remain static. If, due to outside events (codec change, RTP port or IP address change, etc.), the gate parameters become insufficient to carry a forthcoming media stream, the Gate Controller MUST attempt to create a new gate to handle the modified media stream.

8.4.5 Procedures for querying a gate

When a Gate Controller wishes to find out the current parameter settings of a gate, it sends to the CMTS a GATE-INFO message. The CMTS MUST respond to a GATE-INFO message with either a GATE-INFO-ACK (indicating success) or a GATE-INFO-ERR (indicating failure). The Transaction-ID in the response MUST match the transaction ID in the request. GateSpec object(s) MUST be included in the Gate-Info-Ack if they have previously been provided to the CMTS in association with a Gate.

Errors in querying gates are reported by a GATE-INFO-ERR response. The Error object contains one of the following Error-Codes:

2 = Illegal Gate-ID.

127 = Other, unspecified error.

8.4.6 Procedures for committing a gate

When the MTA performs a Commit operation (as described in 6.7 for an RSVP-based MTA, or 7.2.1 for an embedded MTA), the CMTS MUST send a Gate-Open message.

8.4.7 Procedures for closing a gate

The CMTS MUST release all resources associated with the gate, delete the gate, delete associated Service Flow(s) using a DOCSIS DSD message, and send a Gate-Close message when it receives an explicit release message from the MTA client (as described in 6.5.3 for an RSVP-based MTA, or in 7.3.3 for embedded MTAs), or when it detects that the client is no longer actively generating packets and not generating proper refreshes for the flow associated with a gate.

8.4.8 Procedures for deleting a gate

In a normal call flow, a gate is deleted by the CMTS when it receives an RSVP-PATH-TEAR message or the request to release the J.112 Flow via the J.112 MAC-layer Interface (from an embedded MTA that does not support RSVP). The CMTS also deletes a gate at the receipt of a GATE-CLOSE message from a remote CMTS (DCS model) or a CMS (NCS model).

A gate controller, typically, does not initiate a gate delete operation. However, there could be certain abnormal situations where a gate controller might want to delete a gate on the CMTS. For instance, if the gate controller learns (at the receipt of a GATE-ALLOC-ACK response) that a subscriber has exceeded its gate limit, it might want to delete the recently allocated gate at the CMTS. In such scenarios, it SHOULD send a GATE-DELETE message to the CMTS (instead of allowing the gate to time out). There could be other situations in which the delete functionality might be useful.

The CMTS MUST respond to a GATE-DELETE message with a GATE-DELETE-ACK (indicating success) or a GATE-DELETE-ERR (indicating failure). The Transaction-ID in the response MUST match the Transaction-ID in the request. Errors in deleting gates are reported by a GATE-DELETE-ERR response. The Error object contains one of the following Error-Codes:

2 = Illegal Gate-ID.

127 = Other, unspecified error.

8.4.9 Termination sequence

When the CMTS is shutting down its TCP connection to the GC, it MAY first send a DELETE-REQUEST-STATE message (including the handle object used in the REQUEST message). The CMTS MAY follow that with a CLIENT-CLOSE message. These messages are optional because the GC is stateless and because the COPS protocol requires a COPS server to automatically delete any state associated with the CMTS when the TCP connection is terminated.

When the Gate Controller is going to shutdown, it SHOULD send a COPS Client-Close (CC) message to the CMTS. In the COPS CC message, the Gate Controller SHOULD NOT send the PDP redirect address object <PDPRedirAddr>. If the CMTS receives a COPS CC message from the Gate Controller with a <PDPRedirAddr> object, the CMTS MUST ignore the <PDPRedirAddr> while processing the COPS CC message.

8.4.10 Failure scenario

When a CMTS detects the loss of the TCP connection to the GC, e.g., if the GC suffers a catastrophic failure, the CMTS MUST keep all established gates in place. Gates that have been committed will remain committed and gates in any other states will remain in that state until their state is actively changed or the appropriate timers expire. Maintaining gates across GC/CMS failures would allow for any critical flows (e.g., an emergency call) to remain in place.

8.5 CMS use of gate protocol

The CMS MUST ensure that all codecs agreed during negotiation fit within the resource envelope requested from the CMTS using gate communication. CMS MUST use LUB algorithm provided in 6.2.1 to determine b, r, p, m, and M values.

The CMS SHOULD make sure that the Gate Control message communicated to the CMTS contains proper end-point IP addresses and ports such that the call end-points are referenced and possible theft of service is prevented.

The CMS MUST set the Slack Term to a value that is 800 µs for upstream direction if it is not sending an upstream grant jitter parameter to the MTA. Otherwise, the value that is used in the gate should be less than or equal to the value that is sent to the MTA for use as the DOCSIS Tolerated Grant Jitter parameter. For the downstream direction, the CMS MUST set the value to zero.

8.6 Gate-Coordination

The Gate-Controller keeps the state of each gate. The Gate-Controller creates a gate on the CMTS using the Gate-Alloc or Gate-Set message. The Gate-Controller may delete a gate via the Gate-Delete command or may query the CMTS for information associated with a particular gate using the Gate-Info message. The CMTS informs the GC of state changes that occur due to MTA messages or inactivity using the Gate-Open and Gate-Close messages.

The Gate-Open message is generated by the CMTS when the MTA commits QoS resources, thereby starting the call. The Gate-Close message signals the closure of the Gate at the CMTS and the release of associated QoS resources. Both Gate-Open and Gate-Close are informative messages regarding state changes at the CMTS related to a specific Gate, and do not require feedback from the CMS.

The Gate-Open and Gate-Close events at the local and remote end-points must be synchronized to prevent possible theft of service scenarios. This synchronization is accomplished using either CMS internal logic or, in the case of multiple CMS's, using CMS-to-CMS signalling.

8.6.1 Connecting a call

The successful connection of a normal call requires three events happening in close succession:

- the CMS requests commitment of resources at the local MTA;
- the CMTS indicates that resources have been committed by the local MTA;
- coordination of local and remote resource commitment is coordinated on the signalling plane.

See Figure 22.

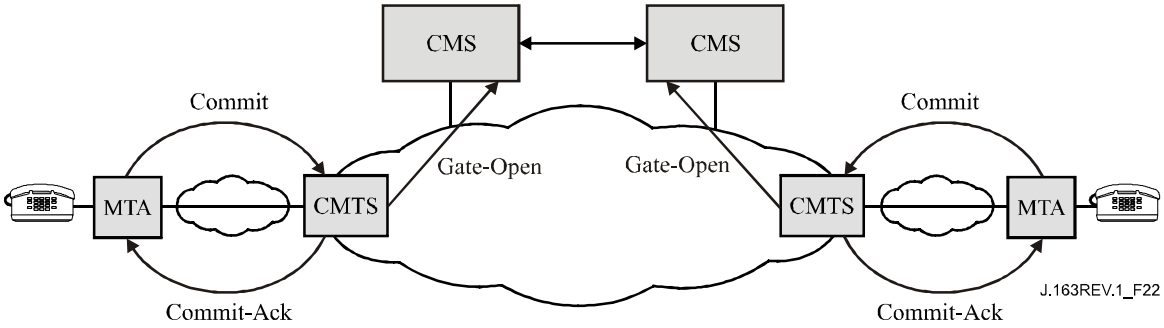


Figure 22/J.163 – Call connection

If a CMS receives a Gate-Open message for a gate that it has not communicated the resources are to be committed then the CMS MUST delete the gate with 'Unexpected Gate-Open' described in the reason-code.

8.6.2 Terminating a call

The termination of a call, as in the case of the connection, requires three events within a short time-frame:

- the CMS requests release of resources at the local MTA;
- the CMTS indicates that resources have been released by the local MTA;
- coordination of local and remote resource release is coordinated on the signalling plane.

See Figure 23.

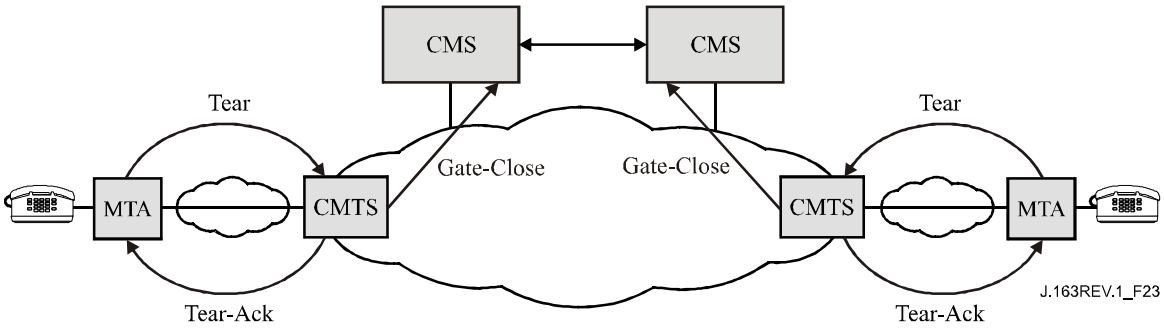


Figure 23/J.163 – Call termination

When the CMS sends the MTA a message to delete the connection, the CMS MUST start a timer for T5 period of time. If by the expiration of the timer the CMTS has not indicated closure of the gate, then the CMS MUST issue a Gate-Delete command to delete the gate at the CMTS with 'Local gate-close failure' described in the reason-code.

When the CMS receives a Gate-Close message, it must update its internal state reflecting the gate removal on the CMTS.

Annex A

Timer definitions and values

Several timers are referenced in this Recommendation. This annex contains the list of those timers, and their recommended values.

Timer-T0

This timer is implemented in the CMTS in the Gate state machine, and limits the period of time that a gate may be allocated without the gate parameters being set. This enables the CMTS to recover the gate-ID resources when the Call Signalling System fails to complete the signalling sequence for a new session.

This timer is started when a gate is allocated.

This timer is reset when the gate parameters are set.

On expiration of this timer, the CMTS MUST consider the assigned gate-ID to be invalid.

The RECOMMENDED value of this timer is 30 seconds.

Timer-T1

This timer is implemented in the CMTS in the Gate state machine, and limits the period of time that may elapse between the authorization and a commit is performed.

This timer is started whenever a Gate is established.

This timer is reset when the Gate goes to a COMMITTED state.

On expiration of this timer, the CMTS MUST release all resources reserved in the CMTS for this gate, revoke any reservations made by the MTA that were authorized by this gate by signalling the CM via a DSC or DSD to release resources it had reserved, and initiate a GATE-CLOSE message for the gate.

Timer-T1 MUST be set to the value given in the GATE-SET message. If the value given in the GATE-SET message is zero, then Timer-T1 MUST be set to a provisionable default value. The RECOMMENDED value of this default is in the range 200-300 seconds.

If the T1 timer value in the Gate-Set is 0, the CMTS MUST return either the provisioned CMTS T1 value or zero for T1 in the GateSpec object of the Gate-Info-Ack message. The provisioned value for T1 is preferred in this case.

Timer-T2

This timer is no longer used.

Timer-T3

This timer is implemented in the MTA or CMTS in the handling of RSVP reservations. It controls the total time that can elapse before the RSVP retransmit process gives up without receiving an acknowledgement in the presence of network loss. It is short enough to recover quickly from lost messages and not significantly impact the post-dial delay, but is long enough to allow the CMTS to acknowledge the request and all intermediate routers in the customer network.

This timer is started when the MTA or CMTS sends an RSVP message that requires an acknowledgement (such as RSVP-PATH). This timer is reset when the sender of the message to be acknowledged receives a response to that message. In the case of an RSVP-PATH message, such a response MAY be RSVP-RESV, RSVP-PATH-ERROR, or RSVP-MESSAGE-ACK, or RSVP-MESSAGE-NACK.

On expiration of this timer, the RSVP retransmit procedure ends.

The RECOMMENDED value of this timer is 4 seconds (4000 ms).

Timer-T4

This timer is implemented in the MTA in the handling of COMMIT messages. It controls the retransmission of COMMIT messages that may have been lost by the network. It is short enough to recover quickly from lost commit requests and not significantly impact the post-pickup delay, but is long enough to allow processing of the COMMIT request at the CMTS.

This timer is started when the MTA sends a COMMIT message.

This timer is reset when the MTA receives a COMMIT-ACK or COMMIT-ERR message that is recognized as a response to the COMMIT.

On expiration of this timer, the MTA re-sends the COMMIT message.

The RECOMMENDED value of this timer is 500 ms.

Timer-T5

This timer is implemented in the CMS. It controls the synchronization between local MTA resource release and CMTS verification of the closure of the local gate.

When the CMS sends the MTA a message to delete the connection, the CMS MUST ensure that the gate is closed in the CMTS within T5. This timer is reset when the CMS receives a confirmation for the local gate closure via the Gate-Close message.

On expiration of this timer, the CMS deletes the gate at the CMTS using Gate-Delete message with 'Local gate-close failure' described in the reason-code.

The RECOMMENDED value of this timer is 5 seconds.

Timer-T6

This timer is implemented in the MTA or CMTS in the handling of RSVP reservations. It controls the initial delay used by the RSVP retransmit procedure.

The RECOMMENDED value of this timer is 500 ms.

Timer-T7

The CMTS MUST set the Timeout for Admitted QoS Parameters for the service flow to the value specified for this timer. The Timeout for Admitted QoS Parameters limits the period of time that the CMTS must hold resources for a service flow's Admitted QoS Parameter Set while they are in excess of its Active QoS Parameter Set. Please see Annex C to Annex B/J.112 for more details on the use of the Timeout for Admitted QoS Parameters.

In order to allow the EMTA to refresh this timer, the CMTS MUST inform the EMTA of the Timeout for Admitted QoS Parameters value in the response (i.e., in the DSA-RSP) to the EMTA's reservation request.

The recommended default value of this timer is 200 seconds.

Timer-T8

The CMTS MUST set the Timeout for Active QoS Parameters for the service flow to the value specified for this timer. The Timeout for Active QoS Parameters limits the period of time resources remain unused on an active service flow. Please see Annex C to Annex B/J.112 for more details on the use of the Timeout for Active QoS Parameters.

In order to allow the EMTA to refresh this timer, the CMTS MUST inform the EMTA of the Timeout for Active QoS Parameters value in the response (i.e., in the DSA-RSP) to the EMTA's reservation request.

The default value of this timer is 0, which instructs the CMTS not to poll for activity on the service flow.

Appendix I

Vacant.

Appendix II

Sample protocol message exchanges for basic DCS on-net to on-net call for stand-alone MTA

This is an informational, informal description of the relationship between the Distributed Call Signalling protocol and the Dynamic QoS methods that may be invoked at different points in the call flow. This description is not meant to be complete. While we attempt to be accurate here in this example, the DCS call signalling specification overrides this description for the specification of the call signalling flows.

When an INVITE message is issued from the originating MTA_O and arrives at the GC_O, the GC_O issues a GATE-ALLOC request to the $CMTS_O$ closest to the originating MTA_O. This is a request for the allocation of a 32-bit GateID that is unique within that $CMTS_O$. This GateID is communicated to the remote $CMTS_T$ in the INVITE message that is forwarded by the GC_O. In addition, the originating $CMTS_O$ communicates the number of active connections (gates) that are used by MTA_O to allow the GC_O or DP to report the current activity level for the subscriber.

The terminating GC_T knows all the possible codecs that may be used for the call, as proposed by MTA_O, and can calculate an "Authorized Envelope" based on this and issue a GATE-SET command to $CMTS_T$. Alternately, GC_T can issue only a GATE-ALLOC command at this time, wait for the results of codec negotiation procedures done by MTA_T, calculate a more accurate "Authorization Envelope" after receiving the 200-OK from MTA_T, and then issue the GATE-SET command. The latter is shown in the following call flow diagrams. In either case, the GateID is allocated and given to MTA_T in the INVITE message, and MTA_T waits for the ACK signalling message to determine the final negotiated codec values.

Included in the 200-OK message from GC_T to GC_O is the GateID at the terminating end. This is provided to $CMTS_O$ in the corresponding GATE-SET exchange along with the "Authorized Envelope" of Flowspec parameters.

After the 200-OK is returned to MTA_O, it knows the address of the destination MTA_T and the parameters associated with the call (codecs used), and translates these to Flowspec parameters for both directions. The originating MTA_O sends out an ACK for the 200-OK and now performs a resource reservation. When the ACK arrives at the terminating MTA_T, it has all the information necessary, and performs a resource reservation.

Reservation involves issuing a RSVP-PATH message with Flowspec parameters for both directions. The CMTS performs admission control, after checking the parameters against both the Authorized Envelope as well as resource availability, and acknowledges successful reservation with a RSVP-RESV message. In between, the J.112 MAC message exchange for the layer 2 resource allocation is performed by the CMTS and the CM. The resources required for the call are now ready to be committed. However, they await one more phase of the call signalling protocol, and the users on both ends of the call picking up the "phone" to communicate.

The second 200-OK message from MTA_T to the originating MTA_O is an indication that the two users (in this simple 2-party basic call) are ready to communicate. The terminating MTA_T sends a COMMIT message immediately after sending the 200-OK. The originating MTA_O on receiving the 200-OK acknowledges this message and issues a COMMIT message also. The COMMIT message

goes from each MTA to its local CMTS, and causes a J.112 MAC message exchange to commit the resources to the flow. When the COMMIT is acknowledged by the CMTSs, the two ends may begin to communicate while receiving enhanced QoS. When the COMMIT message is received by either of the two CMTSs, it starts Timer-T2 that awaits reception of the Gate-Open message from the remote CMTS with its GateID.

Also indicated are the Gate Coordination messages between the two CMTSs indicating to each other that the Gate has been opened, and the description (FlowSpec) of the flow expected from the other end has been exchanged. Reception of the Gate-Open message indicates that the timer at the CMTSs would be disabled.

On completion of the call, the MTAs send a RSVP-PATH-TEAR message to tear down the reservations. At this time, the CMTS also sends a Gate-Close coordination message to the remote CMTS.

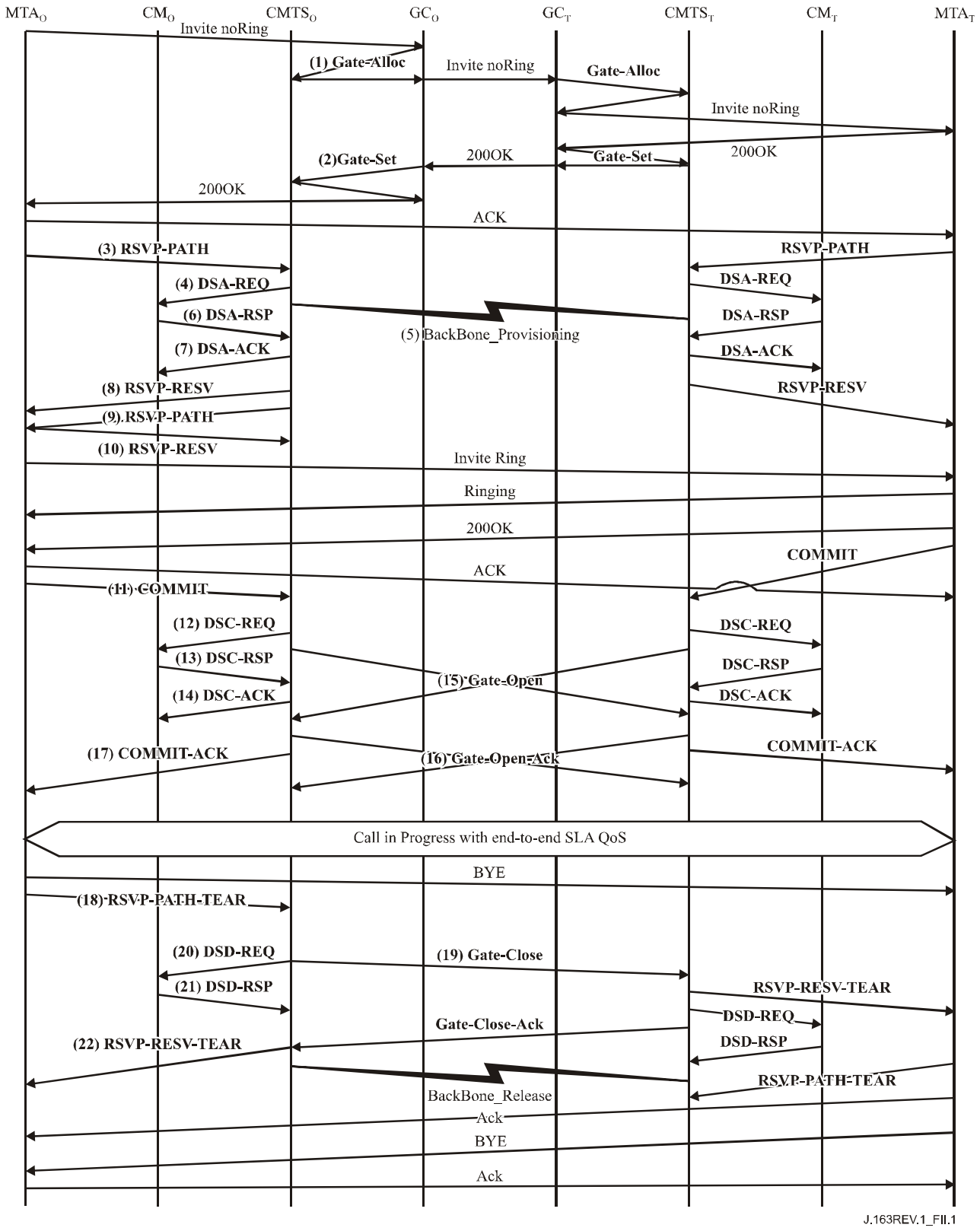


Figure II.1/J.163 – Basic call flow – DCS signalling

1) GCo, upon receipt of signalling information from MTAo, checks the current resource consumption of MTAo by consulting CMTSo.

GATE-ALLOC

TransactionID		3176	
Subscriber		MTAo	Request for total resources in use by this endpoint.
Activity-Count		4	Maximum connections allowed by client.

CMTSo checks current resource usage by MTAo, and responds telling the number of connections active.

GATE-ALLOC-ACK

TransactionID		3176	
Subscriber		MTAo	Request for total resources in use by this endpoint.
Gate-ID		37125	Identifier for allocated Gate.
Activity Count		3	Total connections established by this client.
Gate Coordination Port		4104	UDP port at which CMTS will listen for gate coordination messages.

2) GCo, upon further signalling exchanges, gives CMTSo authorization to admit the new connection.

GATE-SET

TransactionID		3177	Unique Transaction ID for this message exchange.
Subscriber		MTAo	Request for total resources in use by this client.
Gate-ID		37125	Identifier for allocated Gate.
Remote-Gate-Info	CMTS Address	CMTSt	Information needed to perform gate coordination.
	CMTS Port	2052	
	Remote Gate-ID	1273	
	Security Key	<key>	
Event-Generation-Info	RKS-Addr	RKS	Address of Record Keeping Server.
	RKS-Port	3288	Port on Record Keeping Server.
	Billing Correlation ID	<id>	Opaque data that will be passed to RKS when resources are committed.

GATE-SET

Media-Connection-Info	Called-Number	212-555-2222	Fields needed for generation of Call-Answer message.
	Routing Number	212-555-2222	
	Charged Number	212-555-1111	
	Location Routing Number	212-555-2222	
Gate-Spec	Direction	up	
	Protocol	UDP	The protocol, Destination Address, Source Address, and Destination Port quadruple are used for QoS classifiers.
	Source Address	MTAo	
	Destination Address	MTAt	
	Source port	0	
	Destination port	7000	
	DSCP	6	Packet Type value for upstream packets.
	T1	180000	Maximum time between reserve and commit.
	T2	2000	Maximum time for gate coordination to complete.
	r	12000	These are the maximum bandwidth parameters that MTAo is authorized to request for this conversation.
	b	120	
	p	12000	
	m	120	
	M	120	
	R	12000	
	S	0	

GATE-SET

Gate-Spec	Direction	down	
	Protocol	UDP	The protocol, Destination Address, Source Address, and Destination Port quadruple are used for QoS classifiers.
	Source Address	MTAt	
	Destination Address	MTAo	
	Source port	0	
	Destination port	7120	
	DSCP	9	Packet Type value for downstream packets.
	T1	180000	Maximum time between reserve and commit.
	T2	2000	Maximum time for gate coordination to complete.
	r	12000	These are the maximum bandwidth parameters that MTAo is authorized to request for this conversation.
	b	120	
	p	12000	
	m	120	
	M	120	
	R	12000	
	S	0	

CMTSo responds to the Gate Setup command with an acknowledgement.

GATE-SET-ACK

TransactionID		3177	
Subscriber		MTAo	Request for total resources in use by this client.
Gate-ID		37125	Identifier for allocated Gate.
Activity Count		4	Total connections established by this client.

3) MTAo, upon receiving call signalling information, sends an RSVP-PATH message, addressed to MTAt, but with the Router-Alert bit set in the IP header. Intermediate routers in the home LAN intercept, process, and forward this message as a normal RSVP-PATH.

RSVP-PATH

Session-Object	Protocol	UDP	The parameters identify the RSVP session, match the authorization previously sent by the GateController, and are also used for QoS classifiers.
	Destination Address	MTAt	
	Destination port	7000	
Sender Templ	Source Address	MTAo	
	Source port	7120	
Sender-Tspec	r	12000	These are the negotiated traffic parameters actually being requested for this call. The CMTS calculates the actual upstream QoS parameters using these Tspec and Rspec parameters. This is a standard RSVP object, which will be interpreted by all intermediate routers in the path between the MTA and CMTS.
	b	120	
	p	12000	
	m	120	
	M	120	
	Hdr Suppression	40	
	VAD	off	
Forward Rspec	R	12000	
	S	0	
Reverse-Session	Protocol	UDP	New RSVP objects that provides the CMTS with sufficient information to calculate downstream traffic parameters and to generate an RSVP-PATH message for the downstream flow.
	Destination Addr	MTAo	
	Destination port	7120	
Reverse-Sender Templ	Source Address	MTAt	
	Source port	0	
Reverse-Sender-Tspec	r	12000	Negotiated traffic parameters actually being requested for this call. The CMTS calculates the actual downstream QoS parameters using these Tspec and Rspec parameters. This is a new RSVP object, which will be ignored by intermediate routers.
	b	120	
	p	12000	
	m	120	
	M	120	
	Hdr Suppression	0	
	VAD	off	
Reverse-Rspec	R	12000	
	S	0	
Gate-ID		37125	

4) The CMTS uses the RSVP-PATH message and calculates the QoS parameters for the J.112 link. The CMTS sends the following DSA-REQ to the CM. This message is used to establish both upstream and downstream parameters. The Upstream Unsolicited Grant Size was calculated as 120 (from Tspec) plus 18 (Ethernet overhead) minus 40 (Header Suppression amount) plus 13 (J.112 overhead). Header suppression, being specified as length 40 in the RSVP-PATH, indicates the 42 bytes of Ethernet/IP/UDP header. Contents of the suppressed header is taken from the RSVP packet.

DSA-REQ

TransactionID		1
UpstreamServiceFlow	ServiceFlowIdentifier	1001
	QoSParameterSetType	Admitted (2)
	TimeOutAdmitted	200
	ServiceFlowScheduling	UGS (6)
	Request/Transmission Policy	0x00000017
	NominalGrantInterval	10 ms
	ToleratedGrantJitter	2 ms
	GrantsPerInterval	1
	UnsolicitedGrantSize	111
DownstreamServiceFlow	ServiceFlowIdentifier	2001
	QoSParameterSetType	Admitted (2)
	TimeOutAdmitted	200
	TrafficPriority	5
	MaximumSustainedRate	12 000
UpstreamPacketClassification	ServiceFlowIdentifier	1001
	PacketClassifierIdentifier	3001
	ClassifierPriority	150
	ClassifierActivationState	Inactive (0)
	IPSourceAddress	MTAo
	IPSourcePort	7120
	IPDestinationAddress	MTAt
	IPDestinationPort	7000
	IPProtocol	UDP (17)
DownstreamPacketClassification	ServiceFlowIdentifier	2001
	PacketClassifierIdentifier	3002
	ClassifierPriority	150
	ClassifierActivationState	Inactive (0)
	IPSourceAddress	MTAt
	IPDestinationAddress	MTAo
	IPDestinationPort	7120
	IPProtocol	UDP (17)

DSA-REQ

PayloadHeaderSuppression	ClassifierIdentifier	3001
	ServiceFlowIdentifier	1001
	HeaderSuppressionIndex	1
	HeaderSuppressionField	<42bytes>
	HeaderSuppressionMask	<42bits>
	HeaderSuppressionSize	42
	HeaderSuppressionVerify	Verify (0)
HMAC		

5) Simultaneous with message No. 4, the CMTS initiates any required backbone reservations for the requested quality of service. The content of this message is dependent on the particular backbone algorithms in use, and is outside the scope of this Recommendation. The backbone router sends to the CMTS any required notification that the reservation is successful.

6) The CM checks the resources it is required to allocate (e.g., header suppression table space, Service Flow IDs, classifier table space, local network bandwidth), and installs the classifiers. If the operation is successful it returns the DSA-RSP message stating the success.

DSA-RSP

TransactionID		1
ConfirmationCode		Success (0)
HMAC		

7) Upon receipt of the DSA-RSP, the CMTS acknowledges receipt with a DSA-ACK message.

DSA-ACK

TransactionID		1
ConfirmationCode		Success (0)
HMAC		

8) Once the J.112 reservation is complete, and the backbone reservation is successful, the CMTS responds to the RSVP-PATH message by sending an RSVP-RESV message. The message includes the ResourceID that is assigned by the CMTS to this connection. The RSVP-RESV message is sent with the source address of MTAt and destination address of MTAo. All intermediate routers will intercept, process, and forward this as a standard RSVP-RESV message.

RSVP-RESV

Session-Object	Protocol	UDP	These fields identify the IP flow for which the reservation is being established.
	Destination Address	MTAt	
	Destination port	7000	
Flowspec	r	12000	These fields identify the resources being reserved for this flow.
	b	120	
	p	12000	
	m	120	
	M	120	
	R	12000	
	S	0	
ResourceID		1	New Resource ID created for this reservation.

9) If the address of the previous hop differs from the Source Address, then the CMTS is required to generate a RSVP-PATH message to reserve downstream resources at all intermediate routers. This condition would only be met it the MTA was not immediately adjacent to the CM.

For this example, assume an intermediate router exists between MTAo and its CM, but not between MTAt and its CM.

The CMTS constructs RSVP-PATH message using the Reverse Path info it received from the RSVP-PATH message and sends the message to the originating MTA. The message includes the ResourceID object.

RSVP-PATH

Session-Object	Protocol	UDP	The Session-Object and Sender-Template are faked as if the RSVP message had come from the far end.
	Destination Address	MTAo	
	Destination port	7120	
Sender-Tspec	r	12000	The Sender-Tspec came from the Reverse-Sender-Tspec in the RSVP-PATH message from MTAo. This identifies the resources that will be needed in the downstream direction (from MTAt to MTAo).
	b	120	
	p	12000	
	m	120	
	M	120	
	Hdr Suppression	40	
	VAD	off	
Forward Rspec	R	12000	
	S	0	
ResourceID		1	New Resource ID created for this reservation.

10) MTAo, in response to the RSVP-PATH(9), sends RSVP-RESV to MTAt. This message is sent with "router alert" set, and all intermediate routers intercept, process, and forward this message until it reaches the CMTS.

RSVP-RESV

Session-Object	Protocol	UDP	The Session-Object and Sender-Template are copied from the RSVP-PATH message received.
	Destination Address	MTAo	
	Destination port	7120	
Filter-Spec	Source Address	MTAt	
	Source port	7000	
Flowspec	r	12000	These also are copied from the RSVP-PATH message, and specify the amount of resources being reserved for the flow.
	b	120	
	p	12000	
	m	120	
	M	120	
	Hdr Suppression	40	
	VAD	off	
	R	12000	
	S	0	
ResourceID		1	Resource ID, copied from RSVP-PATH.

11) In response to signalling messages that indicate the call has completed (i.e., the other side has gone off-hook), MTAo sends the COMMIT message to the CMTS. This message is directed to the CMTS at a UDP port determined via call signalling.

The Session-Object and Sender Template give the CMTS enough information to identify the "gate" and to identify which reserved resources are being committed.

COMMIT

Session-Object	Protocol	UDP	The protocol, Destination Address, Source Address, and Destination Port quadruple must match those for the Gate ID.
	Destination Address	MTAt	
	Destination port	7000	
Sender Templ	Source Address	MTAo	
	Source port	7120	
Gate-ID		37125	

12) The CMTS resolves which reservation is to be activated, and sends a DSC-REQ to the CM to activate the flow.

DSC-REQ

TransactionID		2
UpstreamServiceFlow	ServiceFlowIdentifier	1001
	QoSParameterSetType	Admitted + Activated (6)
	TimeOutActive	10
	ServiceFlowScheduling	UGS (6)
	Request/TransmissionPolicy	0x00000017
	NominalGrantInterval	10 ms
	ToleratedGrantJitter	2 ms
	GrantsPerInterval	1
	UnsolicitedGrantSize	111
DownstreamServiceFlow	ServiceFlowIdentifier	2001
	QoSParameterSetType	Admitted + Activated (6)
	TimeOutActive	10
	TrafficPriority	5
	MaximumSustainedRate	12 000
UpstreamPacketClassification	ServiceFlowIdentifier	1001
	PacketClassifierIdentifier	3001
	ClassifierChangeAction	Replace (1)
	ClassifierPriority	150
	ClassifierActivationState	Active (1)
	IPSourceAddress	MTAo
	IPSourcePort	7120
	IPDestinationAddress	MTAt
	IPDestinationPort	7000
	IPProtocol	UDP (17)
DownstreamPacketClassification	ServiceFlowIdentifier	2001
	PacketClassifierIdentifier	3002
	ClassifierChangeAction	Replace (1)
	ClassifierPriority	150
	ClassifierActivationState	Active (1)
	IPSourceAddress	MTAt
	IPSourcePort	7000
	IPDestinationAddress	MTAo
	IPDestinationPort	7124
	IPProtocol	UDP (17)
HMAC		

13) The CM sends a DSC-RSP message showing the operation was successful.

DSC-RSP

TransactionID		2
ConfirmationCode		Success (0)
HMAC		

14) The CMTS sends DSC-ACK message to indicate that the DSC-RSP has been received and agreed.

DSC-ACK

TransactionID		2
ConfirmationCode		Success (0)
HMAC		

15) The CMTS sends the gate coordination message to the remote CMTS to inform it that the resources at this end have been committed.

GATE-OPEN

Transaction ID		72	Identifier to match this message with its response.
Gate ID		1273	Gate-ID at remote CMTS.
Tspec	r	12000	These are the committed traffic parameters actually being utilized in the MTAo to MTAt direction.
	b	120	
	p	12000	
	m	120	
	M	120	
Reverse-Tspec	r	12000	These are the expected traffic parameters being utilized in the MTAt to MTAo direction.
	b	120	
	p	12000	
	m	120	
	M	120	
HMAC			Security checksum for this message.

16) The remote CMTS responds to the GATE-OPEN with:

GATE-OPEN-ACK

TransactionID		72	Identifier to match this message with its response.
HMAC			Security checksum for this message.

17) The CMTS acknowledges the COMMIT with:

COMMIT-ACK

Session-Object	Protocol	UDP	The protocol, Destination Address, Source Address, and Destination Port may assist in matching the acknowledgement to the COMMIT message.
	Destination Address	MTAt	
	Destination port	7000	
Sender Templ	Source Address	MTAo	
	Source port	7120	
Gate-ID		37125	

18) When the call is finished the MTA sends RSVP-PATH-TEAR message to the CMTS. For each RSVP reservation, the MTA sends a separate RSVP-PATH-TEAR message.

RSVP-PATH-TEAR

Session-Object	Protocol	UDP	The protocol, Destination Address, Source Address, and Destination Port identify the RSVP flow.
	Destination Address	MTAt	
	Destination port	7000	
Sender Templ	Source Address	MTAo	
	Source port	7120	

19) The CMTS, upon receiving RSVP-PATH-TEAR, sends the gate coordination message to its corresponding CMTS serving MTAt.

GATE-CLOSE

TransactionID		73	Identifier to match this message with its response.
Gate-ID		1273	This identifies the GateID at the remote CMTS.
HMAC			Security checksum for this message.

The remote CMTS responds with:

GATE-CLOSE-ACK

TransactionID		73	Identifier to match this message with its response.
HMAC			Security checksum for this message.

20) The CMTS, upon receiving RSVP-PATH-TEAR, sends a DSD-REQ to the CM indicating the Service Flow ID that is to be deleted.

DSD-REQ

TransactionID		3
ServiceFlowID		1001
HMAC		

DSD-REQ

TransactionID		4
ServiceFlowID		2001
HMAC		

21) The CM deletes the Service Flow ID and sends the response to the CMTS.

DSD-RSP

TransactionID		3
ServiceFlowID		1001
ConfirmationCode		Success (0)
HMAC		

DSD-RSP

TransactionID		4
ServiceFlowID		2001
ConfirmationCode		Success (0)
HMAC		

22) The CMTS sends the RSVP-RESV-TEAR to MTA.

RSVP-RESV-TEAR

Session-Object	Protocol	UDP	These parameters identify the IP flow that is being terminated.
	Destination Address	MTAt	
	Destination port	7000	
Sender Templ	Source Address	MTAo	
	Source port	7120	

Appendix III

Sample protocol message exchanges for basic NCS on-net to on-net call for stand-alone MTA

This is an informational description of a possible relationship between the call signalling protocol (ITU-T Rec. J.162) and the Dynamic QoS methods that may be invoked at various points in the call flow.

When the originating MTA_O completes dialling, i.e., the digit map indicates a complete phone number has been entered, the digits are sent to the CMS_O via a Notify message. The CMS_O, in its first step of initiating a new call, tells the MTA_O to create a new inactive connection. The MTA_O allocates a receive port for the media stream, and responds with an ACK message that includes the Session Description listing all the media streams the MTA_O is willing to receive. The CMS_O performs a GATE-ALLOC exchange with the $CMTS_O$ to allocate a Gate-ID, and passes this information to the terminating CMS_T along with the originating SDP profile.

The terminating CMS_T sets up the gate at the terminating $CMTS_T$ (using a GATE-SET command), allowing all of the media flows that are acceptable to the originator within the "Authorized Envelope", and allowing a wildcard destination port on MTA_T. The $CMTS_T$ also assigns a Gate-ID and returns it to the CMS_T. The CMS_T passes the local Gate-ID to the terminating MTA_T in a Create Connection command, along with the proposed SDP profile. MTA_T, in its response, indicates the set of media streams it finds acceptable, and the allocated port for reception of those streams.

At this point, MTA_T knows the sending codec, the receiving codec, the destination address and port for voice packets it sends, and the local port for reception of voice packets. It therefore begins the reserve sequence by sending a RSVP-PATH to $CMTS_T$.

When CMS_o receives the SDP profile from MTA_T, it has sufficient information to establish the gate at $CMTS_O$. It therefore performs the GATE-SET operation, including the remote Gate-ID and the address of $CMTS_T$. CMS_O now issues a Modify Connection command to MTA_O, telling it the destination address, port, and codec to use. MTA_O now has sufficient information to make a resource reservation. When the reservation completes, it sends a successful acknowledgement to CMS_O. CMS_T now tells MTA_T to alert the user of an incoming call. MTA_T first checks that the resource reservation that it earlier initiated has completed successfully, and if so, proceeds to ring the phone.

When the called party answers, MTA_T informs CMS_T with a Notify message, indicating Offhook. CMS_T now sends a Modify Connection command to MTA_T making the connection mode send+receive; MTA_T does the COMMIT exchange with $CMTS_T$ and then sends the acknowledgement. CMS_O also sends a Modify Connection command to MTA_O making its connection mode send+receive, causing MTA_O to also do the COMMIT exchange with $CMTS_O$. The call is now established.

Either party can initiate a call termination by sending a Notify message to their CMS indicating Onhook. In the diagram, MTA_O is shown doing this. CMS_O responds to the Onhook notification by sending a Delete Connection command, which triggers the RSVP-PATH-TEAR sequence to release the resources. MTA_T is informed of the hangup both by call signalling (a Delete Connection command, not shown) or by the RSVP-RESV-TEAR DQoS message. When MTA_T later goes onhook, it produces the same Notify message as was earlier sent by MTA_O, and ends the sequence. See Figure III.1.

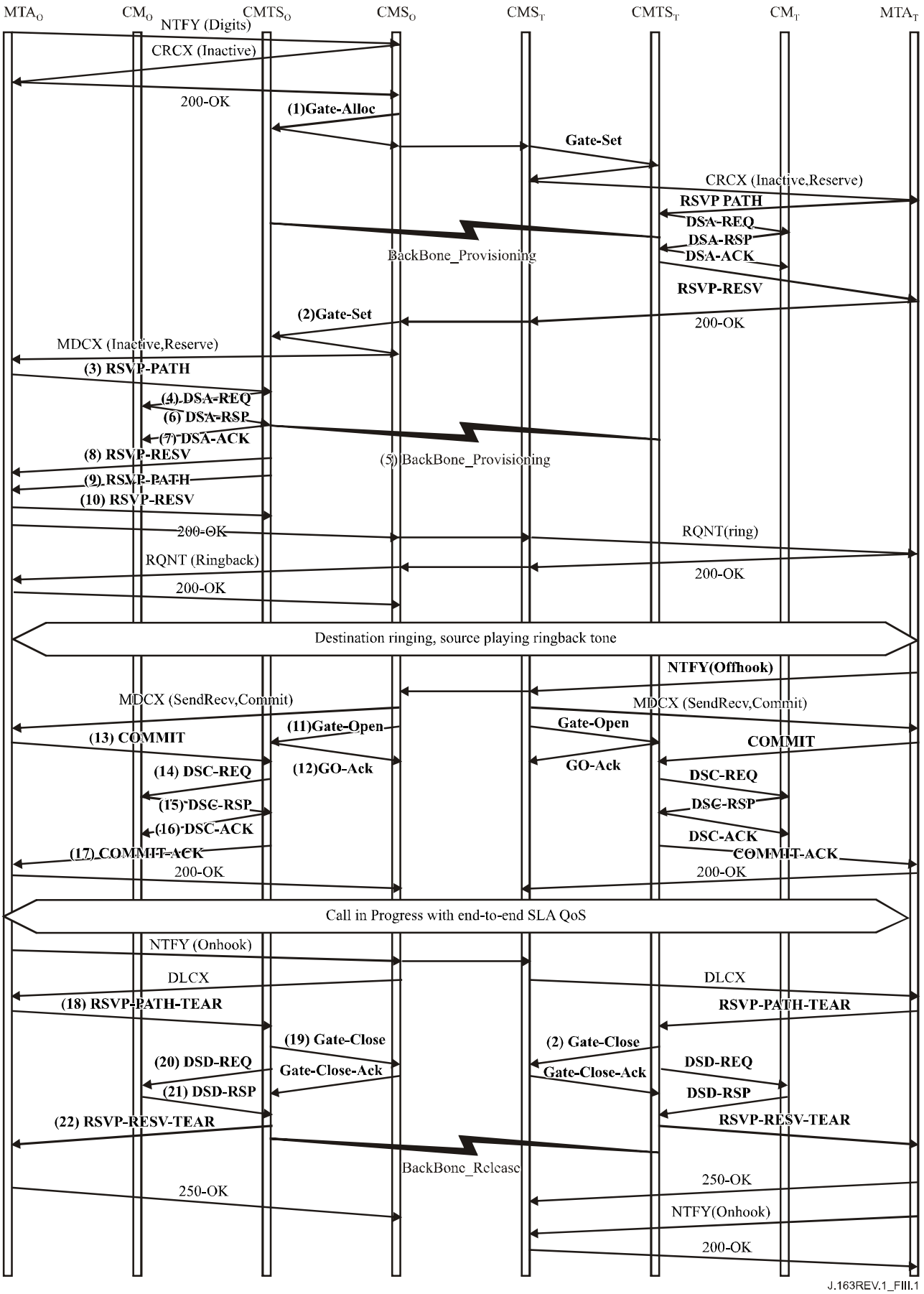


Figure III.1/J.163 – Basic call flow – NCS

1) CMSo, upon receipt of signalling information from MTAo, checks the current resource consumption of MTAo by consulting CMTSo.

GATE-ALLOC

TransactionID		3176	
Subscriber		MTAo	Request for total resources in use by this endpoint.
Activity-Count		12	Maximum connections allowed by client.

CMTSo checks current resource usage by MTAo, and responds telling the number of connections active.

GATE-ALLOC-ACK

TransactionID		3176	
Subscriber		MTAo	Request for total resources in use by this endpoint.
Gate-ID		37125	Identifier for allocated Gate.
Activity Count		3	Total connections established by this client.

2) CMSo, upon further signalling exchanges, gives CMTSo authorization to admit the new connection.

GATE-SET

TransactionID		3177	Unique Transaction ID for this message exchange.
Subscriber		MTAo	Request for total resources in use by this client.
Gate-ID		37125	Identifier for allocated Gate.
Remote-Gate-Info	Address	CMSo	Information needed to perform gate coordination. Note that CMSo has given itself as the entity for exchanging gate coordination messages. Flag value indicates that the CMTS should not send a Gate-Open message when it receives a COMMIT from the MTA, but still expect to receive a Gate-Open message from CMSo.
	Port	2052	
	Remote Gate-ID	8095	
	Security Key	<key>	
	Flag	No-Send-Gate-Open	
Event-Generation-Info	RKS-Addr	RKS	Address of Record Keeping Server.
	RKS-Port	3288	Port on Record Keeping Server.
	Billing Correlation ID	<id>	Opaque data that will be passed to RKS when resources are committed.

GATE-SET

Gate-Spec	Direction	up	
	Protocol	UDP	The protocol, Destination Address, Source Address, and Destination Port quadruple are used for QoS classifiers.
	Source Address	MTAo	
	Destination Address	MTAt	
	Source port	0	
	Destination port	7000	
	DSCP	6	Packet Type value for upstream packets.
	T1	180000	Maximum time between reserve and commit.
	T2	2000	Maximum time for gate coordination to complete.
	r	12000	These are the maximum bandwidth parameters that MTAo is authorized to request for this conversation.
	b	120	
	p	12000	
	m	120	
	M	120	
	R	12000	
	S	0	
Gate-Spec	Direction	down	
	Protocol	UDP	The protocol, Destination Address, Source Address, and Destination Port quadruple are used for QoS classifiers.
	Source Address	MTAt	
	Destination Address	MTAo	
	Source port	0	
	Destination port	7120	
	DSCP	9	Packet Type value for downstream packets.
	T1	180000	Maximum time between reserve and commit.
	T2	2000	Maximum time for gate coordination to complete.
	r	12000	These are the maximum bandwidth parameters that MTAo is authorized to request for this conversation.
	b	120	
	p	12000	
	m	120	
	M	120	
	R	12000	
	S	0	

CMTSo responds to the Gate Setup command with an acknowledgement.

GATE-SET-ACK

TransactionID		3177	
Subscriber		MTAo	Request for total resources in use by this client.
Gate-ID		37125	Identifier for allocated Gate.
Activity Count		3	Total connections established by this client.

3) MTAo, upon receiving a Modify-Connection command, sends an RSVP-PATH message, addressed to MTAt, but with the Router-Alert bit set in the IP header. Intermediate routers in the home LAN intercept, process, and forward this message as a normal RSVP-PATH.

RSVP-PATH

Session-Object	Protocol	UDP	The parameters identify the RSVP session, match the authorization previously sent by the GateController, and are also used for QoS classifiers.
	Destination Address	MTAt	
	Destination port	7000	
Sender Templ	Source Address	MTAo	
	Source port	7120	
Sender-Tspec	r	12000	These are the negotiated traffic parameters actually being requested for this call. The CMTS calculates the actual upstream QoS parameters using these Tspec and Rspec parameters. This is a standard RSVP object, which will be interpreted by all intermediate routers in the path between the MTA and CMTS.
	b	120	
	p	12000	
	m	120	
	M	120	
	Hdr Suppression	40	
	VAD	off	
Forward Rspec	R	12000	
	S	0	
Reverse-Session	Protocol	UDP	New RSVP objects that provide the CMTS with sufficient information to calculate downstream traffic parameters and to generate an RSVP-PATH message for the downstream flow.
	Destination Addr	MTAo	
	Destination port	7120	
Reverse-Sender Templ	Source Address	MTAt	
	Source port	0	
Reverse -Sender-Tspec	r	12000	Negotiated traffic parameters actually being requested for this call. The CMTS calculates the actual downstream QoS parameters using these Tspec and Rspec parameters. This is a new RSVP object, which will be ignored by intermediate routers.
	b	120	
	p	12000	
	m	120	
	M	120	
	Hdr Suppression	0	
	VAD	off	
Reverse-Rspec	R	12000	
	S	0	
Gate-ID		37125	

4) The CMTS uses the RSVP-PATH message and calculates the QoS parameters for the J.112 link. The CMTS sends the following DSA-REQ to the CM. This message is used to establish both upstream and downstream parameters. The Upstream Unsolicited Grant Size was calculated as 120 (from Tspec) plus 18 (Ethernet overhead) minus 40 (Header Suppression amount) plus 13 (J.112 overhead). Header suppression, being specified as length 40 in the RSVP-PATH, indicates the 42 bytes of Ethernet/IP/UDP header. Contents of the suppressed header is taken from the RSVP packet.

DSA-REQ

TransactionID		1
UpstreamServiceFlow	ServiceFlowIdentifier	1001
	QoSParameterSetType	Admitted (2)
	TimeOutAdmitted	200
	ServiceFlowScheduling	UGS (6)
	Request/TransmissionPolicy	0x00000017
	NominalGrantInterval	10 ms
	ToleratedGrantJitter	2 ms
	GrantsPerInterval	1
	UnsolicitedGrantSize	111
DownstreamServiceFlow	ServiceFlowIdentifier	2001
	QoSParameterSetType	Admitted (2)
	TimeOutAdmitted	200
	TrafficPriority	5
	MaximumSustainedRate	12000
UpstreamPacketClassification	ServiceFlowIdentifier	1001
	PacketClassifierIdentifier	3001
	ClassifierPriority	150
	ClassifierActivationState	Inactive (0)
	IPSourceAddress	MTAo
	IPSourcePort	7120
	IPDestinationAddress	MTAt
	IPDestinationPort	7000
	IPProtocol	UDP (17)
DownstreamPacketClassification	ServiceFlowIdentifier	2001
	PacketClassifierIdentifier	3002
	ClassifierPriority	150
	ClassifierActivationState	Inactive (0)
	IPSourceAddress	MTAt
	IPSourcePort	7000
	IPDestinationAddress	MTAo
	IPDestinationPort	7120
	IPProtocol	UDP (17)

DSA-REQ

PayloadHeaderSuppression	ClassifierIdentifier	3001
	ServiceFlowIdentifier	1001
	HeaderSuppressionIndex	1
	HeaderSuppressionField	<42bytes>
	HeaderSuppressionMask	<42bits>
	HeaderSuppressionSize	42
	HeaderSuppressionVerify	Verify (0)
HMAC		

5) Simultaneous with message No. 4, the CMTS initiates any required backbone reservations for the requested quality of service. The content of this message is dependent on the particular backbone algorithms in use, and is outside the scope of this Recommendation. The backbone router sends to the CMTS any required notification that the reservation is successful.

6) The CM checks the resources it is required to allocate (e.g., header suppression table space, Service Flow IDs, classifier table space, local network bandwidth), and installs the classifiers. If the operation is successful it returns the DSA-RSP message stating the success.

DSA-RSP

TransactionID		1
ConfirmationCode		Success (0)
HMAC		

7) Upon receipt of the DSA-RSP, the CMTS acknowledges receipt with a DSA-ACK message.

DSA-ACK

TransactionID		1
ConfirmationCode		Success (0)
HMAC		

8) Once the J.112 reservation is complete, and the backbone reservation is successful, the CMTS responds to the RSVP-PATH message by sending an RSVP-RESV message. The message includes the ResourceID that is assigned by the CMTS to this connection. The RSVP-RESV message is sent with the source address of MTAt and destination address of MTAo. All intermediate routers will intercept, process, and forward this as a standard RSVP-RESV message.

RSVP-RESV

Session-Object	Protocol	UDP	These fields identify the IP flow for which the reservation is being established.
	Destination Address	MTAt	
	Destination port	7000	
Filter-Spec	Source Address	MTAo	
	Source port	7120	

RSVP-RESV

Flowspec	r	12000	These fields identify the resources being reserved for this flow.
	b	120	
	p	12000	
	m	120	
	M	120	
	R	12000	
	S	0	
ResourceID		1	New Resource ID created for this reservation.

9) If the address of the previous hop differs from the Source Address, then the CMTS is required to generate a RSVP-PATH message to reserve downstream resources at all intermediate routers. This condition would only be met it the MTA was not immediately adjacent to the CM.

For this example, assume an intermediate router exists between MTAo and its CM, but not between MTAt and its CM.

The CMTS constructs RSVP-PATH message using the Reverse Path info it received from the RSVP-PATH message and sends the message to the originating MTA. The message includes the ResourceID object.

RSVP-PATH

Session-Object	Protocol	UDP	The Session-Object and Sender-Template are faked as if the RSVP message had come from the far end.
	Destination Address	MTAo	
	Destination port	7120	
Sender-Tspec	r	12000	The Sender-Tspec came from the Reverse-Sender-Tspec in the RSVP-PATH message from MTAo. This identifies the resources that will be needed in the downstream direction (from MTAt to MTAo).
	b	120	
	p	12000	
	m	120	
	M	120	
	Hdr Suppression	40	
	VAD	off	
Forward Rspec	R	12000	
	S	0	
ResourceID		1	New Resource ID created for this reservation.

10) MTAo, in response to the RSVP-PATH (9), sends RSVP-RESV to MTAt. This message is sent with "router alert" set, and all intermediate routers intercept, process, and forward this message until it reaches the CMTS.

RSVP-RESV

Session-Object	Protocol	UDP	The Session-Object and Sender-Template are copied from the RSVP-PATH message received.
	Destination Address	MTAo	
	Destination port	7120	

RSVP-RESV

Flowspec	r	12000	These also are copied from the RSVP-PATH message, and specify the amount of resources being reserved for the flow.
	b	120	
	p	12000	
	m	120	
	M	120	
	Hdr Suppression	40	
	VAD	off	
	R	12000	
	S	0	
ResourceID		1	Resource ID, copied from RSVP-PATH.

11) The CMS sends the gate coordination message to the CMTS to inform it that the resources should be committed. If the CMTS does not receive a COMMIT message from the MTA within Timer T2, it will abort the connection.

GATE-OPEN

Transaction ID		8096	Identifier to match this message with its response.
GateID		37125	Gate-ID at remote CMTS.
HMAC			Security checksum for this message.

12) The CMTS responds to the GATE-OPEN with:

GATE-OPEN-ACK

TransactionID		8096	Identifier to match this message with its response.
HMAC			Security checksum for this message.

13) In response to Modify-Connection command, which indicates the call has completed (i.e., the other side has gone off-hook), MTAo sends the COMMIT message to the CMTS. This message is directed to the CMTS at a UDP port given in the RSVP-RESV Commit-Entity object. The Session-Object and Sender Template give the CMTS enough information to identify the "gate" and to identify which reserved resources are being committed.

COMMIT

Session-Object	Protocol	UDP	The protocol, Destination Address, Source Address, and Destination Port quadruple must match those for the Gate ID.
	Destination Address	MTAt	
	Destination port	7000	
Sender Templ	Source Address	MTAo	
	Source port	7120	
Gate-ID		37125	

14) The CMTS resolves which reservation is to be activated, and sends a DSC-REQ to the CM to activate the flow.

DSC-REQ

TransactionID		2
UpstreamServiceFlow	ServiceFlowIdentifier	1001
	QoSParameterSetType	Admitted + Activated (6)
	TimeOutActive	10
	ServiceFlowScheduling	UGS (6)
	Request/TransmissionPolicy	0x00000017
	NominalGrantInterval	10 ms
	ToleratedGrantJitter	2 ms
	GrantsPerInterval	1
	UnsolicitedGrantSize	111
DownstreamServiceFlow	ServiceFlowIdentifier	2001
	QoSParameterSetType	Admitted + Activated (6)
	TimeOutActive	10
	TrafficPriority	5
	MaximumSustainedRate	12000
UpstreamPacketClassification	ServiceFlowIdentifier	1001
	PacketClassifierIdentifier	3001
	ClassifierChangeAction	Replace (1)
	ClassifierPriority	150
	ClassifierActivationState	Active (1)
	IPSourceAddress	MTAo
	IPSourcePort	7120
	IPDestinationAddress	MTAt
	IPDestinationPort	7000
	IPProtocol	UDP (17)
DownstreamPacketClassification	ServiceFlowIdentifier	2001
	PacketClassifierIdentifier	3002
	ClassifierChangeAction	Replace (1)
	ClassifierPriority	150
	ClassifierActivationState	Active (1)
	IPSourceAddress	MTAt
	IPSourcePort	7000
	IPDestinationAddress	MTAo
	IPDestinationPort	7124
	IPProtocol	UDP (17)
HMAC		

15) The CM sends a DSC-RSP message showing the operation was successful.

DSC-RSP

TransactionID		2
ConfirmationCode		Success (0)
HMAC		

16) The CMTS sends DSC-ACK message to indicate that the DSC-RSP has been received and agreed.

DSC-ACK

TransactionID		2
ConfirmationCode		Success (0)
HMAC		

17) The CMTS acknowledges the COMMIT with:

COMMIT-ACK

Session-Object	Protocol	UDP	The protocol, Destination Address, Source Address, and Destination Port may assist in matching the acknowledgement to the COMMIT message.
	Destination Address	MTAt	
	Destination port	7000	
Sender Templ	Source Address	MTAo	
	Source port	7120	
Gate-ID		37125	

18) When the call is finished, in response to a Delete-Connection command, the MTA sends RSVP-PATH-TEAR message to the CMTS. For each RSVP reservation, the MTA sends a separate RSVP-PATH-TEAR message.

RSVP-PATH-TEAR

Session-Object	Protocol	UDP	The protocol, Destination Address, Source Address, and Destination Port identify the RSVP flow.
	Destination Address	MTAt	
	Destination port	7000	
Sender Templ	Source Address	MTAo	
	Source port	7120	

19) The CMTS, upon receiving RSVP-PATH-TEAR, sends the gate coordination message to the address given in the GATE-SET command earlier, which in the case of NCS is the Call Agent.

GATE-CLOSE

TransactionID		73	Identifier to match this message with its response.
Gate-ID		8095	This identifies the GateID at the remote CMTS.
HMAC			Security checksum for this message.

The CMS responds with:

GATE-CLOSE-ACK

TransactionID		73	Identifier to match this message with its response.
HMAC			Security checksum for this message.

20) The CMTS, upon receiving RSVP-PATH-TEAR, sends a DSD-REQ to the CM indicating the Service Flow ID that is to be deleted.

DSD-REQ

TransactionID		3
ServiceFlowID		1001
HMAC		

DSD-REQ

TransactionID		4
ServiceFlowID		2001
HMAC		

21) The CM deletes the Service Flow ID and sends the response to the CMTS.

DSD-RSP

TransactionID		3
ServiceFlowID		1001
ConfirmationCode		Success (0)
HMAC		

DSD-RSP

TransactionID		4
ServiceFlowID		2001
ConfirmationCode		Success (0)
HMAC		

22) The CMTS sends the RSVP-RESV-TEAR to MTA.

RSVP-RESV-TEAR

Session-Object	Protocol	UDP	These parameters identify the IP flow that is being terminated.
	Destination Address	MTAt	
	Destination port	7000	
Sender Templ	Source Address	MTAo	
	Source port	7120	

Appendix IV

Sample protocol message exchanges for mid-call codec change

The codec change is achieved by the MTAs transmitting a new RSVP-PATH message subsequent to the call signalling exchange between them to determine what new codec is being used. The new FlowSpec for the call is described in the RSVP-PATH, and must fit within the Authorized Envelope specified in the Gate-Set message that was exchanged between the GCs and the CMTSs earlier for this Gate. The RSVP-PATH includes the same GateID that was previously used for this call. Observe that the initial INVITE to establish the call should have included the codecs in the SDP to ensure that the Authorized Envelope is large enough to accommodate the codec change.

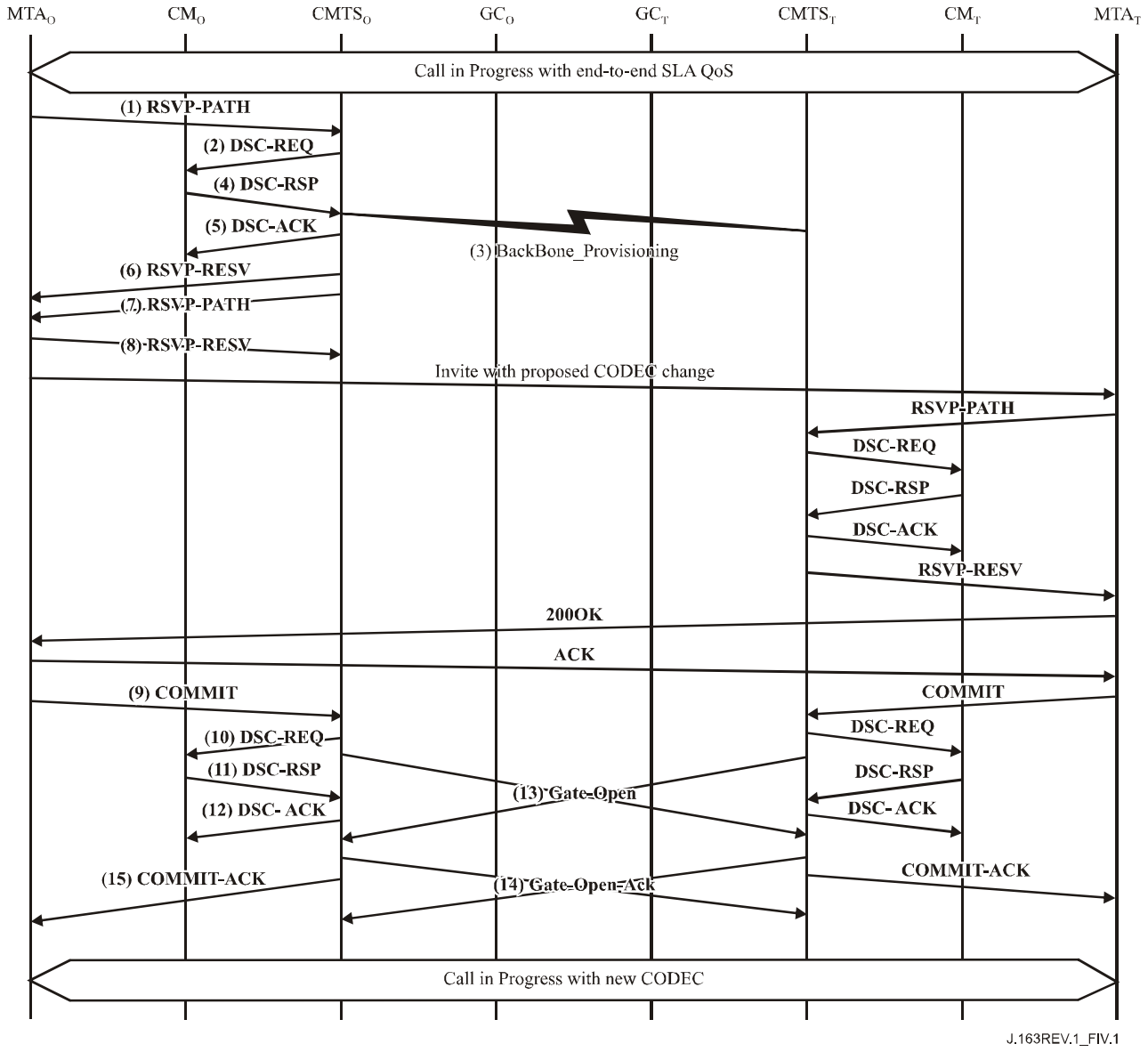


Figure IV.1/J.163 – QoS signalling for codec change

1) MTAo and MTAt are assumed to have a G.728 (20 ms packets, each 80 bytes) call active when MTAo decides, for whatever reason, that a CODEC change is needed to G.711 (10 ms packets, each 120 bytes). After an initial signalling exchange that determines MTAt is capable of handling the desired new CODEC, MTAo sends an RSVP-PATH message addressed to MTAt, but with the Router-Alert bit set in the IP header. Intermediate routers in the home LAN intercept, process, and forward this message as a normal RSVP-PATH, understanding only the least-upper-bound set of traffic parameters given in the Sender-Tspec.

RSVP-PATH

Session-Object	Protocol	UDP	The parameters identify the RSVP session, match the authorization previously sent by the GateController, and are also used for QoS classifiers.
	Destination Address	MTAt	
	Destination port	7000	
Sender Templ	Source Address	MTAo	
	Source port	7120	
Sender-Tspec	r	12000	These give the Least-Upper-Bound for all of the individual traffic parameters for the two separate possible flows. This is a standard RSVP object, which will be interpreted by all intermediate routers in the path between the MTA and CMTS.
	b	120	
	p	12000	
	m	120	
	M	120	
	Hdr Suppression	40	
	VAD	off	
Forward Rspec	R	12000	
	S	0	
Reverse-Session	Protocol	UDP	New RSVP objects that provide the CMTS with sufficient information to calculate downstream traffic parameters and to generate an RSVP-PATH message for the downstream flow.
	Destination Addr	MTAo	
	Destination port	7120	
Reverse-Sender Templ	Source Address	MTAt	
	Source port	7000	
Reverse-Sender-Tspec	r	12000	Negotiated traffic parameters for the new CODEC being requested for this call. The CMTS calculates the actual downstream QoS parameters using these Tspec and Rspec parameters. This is a new RSVP object, which will be ignored by intermediate routers.
	b	120	
	p	12000	
	m	120	
	M	120	
	Hdr Suppression	0	
	VAD	off	
Reverse-Rspec	R	12000	
	S	0	
ResourceID		472	Resource ID assigned for existing call
Gate-ID		37125	Identity of gate that authorizes this request

2) The CMTS uses the RSVP-PATH message and calculates the new QoS parameters for the J.112 link. Since the G.728 stream fits completely within an allocation for G.711, there is no need for a separate Service Flow; therefore the existing Service Flows are modified to increase the admitted bandwidth. The CMTS sends the following DSC-REQ to the CM. This message is used to establish both upstream and downstream parameters. The Upstream Unsolicited Grant Size was calculated as 120 (from Tspec) plus 18 (Ethernet overhead) minus 40 (Header Suppression amount) plus 13 (J.112 overhead). Header suppression, being specified as length 40 in the RSVP-PATH, indicates the 42 bytes of Ethernet/IP/UDP. Contents of the suppressed header are taken from the RSVP packet.

DSC-REQ

TransactionID		1
UpstreamServiceFlow	ServiceFlowIdentifier	1001
	QoSParameterSetType	Admitted (2)
	TimeOutAdmitted	200
	ServiceFlowScheduling	UGS (6)
	Request/TransmissionPolicy	0x00000017
	NominalGrantInterval	10 ms
	ToleratedGrantJitter	2 ms
	GrantsPerInterval	1
	UnsolicitedGrantSize	111
UpstreamServiceFlow	ServiceFlowIdentifier	1001
	QoSParameterSetType	Active (4)
	TimeOutActive	10
	ServiceFlowScheduling	UGS (6)
	NominalGrantInterval	20 ms
	ToleratedGrantJitter	2 ms
	GrantsPerInterval	1
	UnsolicitedGrantSize	71
DownstreamServiceFlow	ServiceFlowIdentifier	2001
	QoSParameterSetType	Admitted (2)
	TimeOutActive	10
	TrafficPriority	5
	MaximumSustainedRate	12000
DownstreamServiceFlow	ServiceFlowIdentifier	2001
	QoSParameterSetType	Active (4)
	TimeOutActive	10
	TrafficPriority	5
	MaximumSustainedRate	4000
HMAC		

3) Simultaneous with message No. 2, the CMTS initiates any required backbone reservations for the requested quality of service. The content of this message is dependent on the particular backbone algorithms in use, and is outside the scope of this Recommendation. The backbone router sends to the CMTS any required notification that the reservation is successful.

4) The CM checks the additional resources it is required to allocate (e.g., local network bandwidth). If the operation is successful, it returns the DSC-RSP message stating the success.

DSC-RSP

TransactionID		1
ConfirmationCode		Success (0)
HMAC		

5) Upon receipt of the DSC-RSP, the CMTS acknowledges receipt with a DSA-ACK message.

DSC-ACK

TransactionID		1
ConfirmationCode		Success (0)
HMAC		

6) Once the J.112 reservation is complete, and the backbone reservation is successful, the CMTS responds to the RSVP-PATH message by sending an RSVP-RESV message. The message includes the Least-Upper-Bound of the two Sender-Tspecs, causing the intermediate routers to allocate resources sufficient to cover either flow. The RSVP-RESV message is sent with the source address of MTAt and destination address of MTAo. All intermediate routers will intercept, process, and forward this as a standard RSVP-RESV message.

RSVP-RESV

Session-Object	Protocol	UDP	These fields identify the IP flow for which the reservation is being established.
	Destination Address	MTAt	
	Destination port	7000	
Filter-Spec	Source Address	MTAo	
	Source port	7120	
Flowspec	r	12000	These fields identify the resources being reserved for this flow. These values are the Least-Upper-Bound of the two Tspecs given in the RSVP-PATH.
	b	120	
	p	12000	
	m	120	
	M	120	
	R	12000	
	S	0	
ResourceID		1	Resource ID previously created for this reservation.

7) If the address of the previous hop differs from the Source Address, then the CMTS is required to generate a RSVP-PATH message to reserve downstream resources at all intermediate routers. This flag would only be set if the MTA was not immediately adjacent to the CM.

The CMTS constructs RSVP-PATH message using the Reverse Path info it received from the RSVP-PATH message and sends the message to the originating MTA. The message includes the ResourceID object.

RSVP-PATH

Session-Object	Protocol	UDP	The Session-Object and Sender-Template are faked as if the RSVP message had come from the far end.
	Destination Address	MTAo	
	Destination port	7120	
Sender Templ	Source Address	MTAt	
	Source port	7000	
Sender-Tspec	r	12000	The Sender-Tspec came from the Reverse-Sender-Tspec in the RSVP-PATH message from MTAo. This identifies the resources that will be needed in the downstream direction (from MTAt to MTAo). This Tspec is the Least-Upper-Bound of the two individual Tspecs sent to the CMTS, causing all intermediate routers to allocate enough resources for either flow.
	b	120	
	p	12000	
	m	120	
	M	120	
	Hdr Suppression	40	
	VAD	off	
Forward Rspec	R	12000	
	S	0	
ResourceID		1	Resource ID previously created for this reservation.

8) MTAo, in response to the RSVP-PATH (7), sends RSVP-RESV to MTAt. This message is sent with "router alert" set, and all intermediate routers intercept, process, and forward this message until it reaches the CMTS.

RSVP-RESV

Session-Object	Protocol	UDP	The Session-Object and Sender-Template are copied from the RSVP-PATH message received.
	Destination Address	MTAo	
	Destination port	7120	
Filter-Spec	Source Address	MTAt	
	Source port	7000	
Flowspec	r	12000	These also are copied from the RSVP-PATH message, and specify the amount of resources being reserved for the flow.
	b	120	
	p	12000	
	m	120	
	M	120	
	Hdr Suppression	40	
	VAD	off	
	R	12000	
	S	0	
ResourceID		1	Resource ID, copied from RSVP-PATH.

9) In response to end-to-end signalling messages that indicate the resource reservation was successful at both ends, MTAo sends the COMMIT message to the CMTS. This message is directed to the CMTS at a UDP port determined via call signalling.

The Session-Object and Sender Template give the CMTS information to verify the Gate-ID and to identify which reserved resources are being committed.

COMMIT

Session-Object	Protocol	UDP	The protocol, Destination Address, Source Address, and Destination Port quadruple must match those for the Gate-ID.
	Destination Address	MTAt	
	Destination port	7000	
Sender Templ	Source Address	MTAo	
	Source port	7120	
Gate-ID		37125	

10) The CMTS resolves which reservation is to be activated, and sends a DSC-REQ to the CM to activate the flow.

DSC-REQ

TransactionID		2
UpstreamServiceFlow	ServiceFlowIdentifier	1001
	QoSParameterSetType	Admitted + Activated (6)
	TimeOutActive	10
	ServiceFlowScheduling	UGS (6)
	Request/TransmissionPolicy	0x00000017
	NominalGrantInterval	10 ms
	ToleratedGrantJitter	2 ms
	GrantsPerInterval	1
	UnsolicitedGrantSize	111
DownstreamServiceFlow	ServiceFlowIdentifier	2001
	QoSParameterSetType	Admitted + Activated (6)
	TimeOutActive	10
	TrafficPriority	5
	MaximumSustainedRate	12000
HMAC		

11) The CM sends a DSC-RSP message showing the operation was successful.

DSC-RSP

TransactionID		2
ConfirmationCode		Success (0)
HMAC		

12) The CMTS sends DSC-ACK message to indicate that the DSC-RSP has been received and agreed.

DSC-ACK

TransactionID		2
ConfirmationCode		Success (0)
HMAC		

13) The CMTS sends the gate coordination message to the remote CMTS to inform it that the resources at this end have been committed.

GATE-OPEN

TransactionID		74	Identifier to match this message with its response.
Gate-ID		1273	Gate-ID at remote CMTS.
Tspec	r	12000	These are the committed traffic parameters actually being utilized in the MTAo to MTAt direction.
	b	120	
	p	12000	
	m	120	
	M	120	
Reverse-Tspec	r	12000	These are the expected traffic parameters being utilized in the MTAt to MTAo direction.
	b	120	
	p	12000	
	m	120	
	M	120	
HMAC			Security checksum for this message.

14) The remote CMTS responds to the GATE-OPEN with:

GATE-OPEN-ACK

Transaction ID		74	Identifier to match this message with its response.
HMAC			Security checksum for this message.

15) The CMTS acknowledges the COMMIT with:

COMMIT-ACK

Session-Object	Protocol	UDP	The protocol, Destination Address, Source Address, and Destination Port may assist in matching the acknowledgement to the COMMIT message.
	Destination Address	MTAt	
	Destination port	7000	
Sender Templ	Source Address	MTAo	
	Source port	7120	
Gate-ID		37125	

Appendix V

Sample protocol message exchanges for Call Hold

Putting a call on Hold at an MTA is performed by sending an INVITE to the MTA with SDP parameters being zero. This results in the MTA sending a COMMIT message with a Flow Spec of 0. Also included is a Resource ID. This enables the CMTS to hold on to the admitted resources, but will now commit zero resources to the flow. This is performed with a MAC message exchange at the J.112 MAC level.

V.1 Example call flow

See Figure V.1.

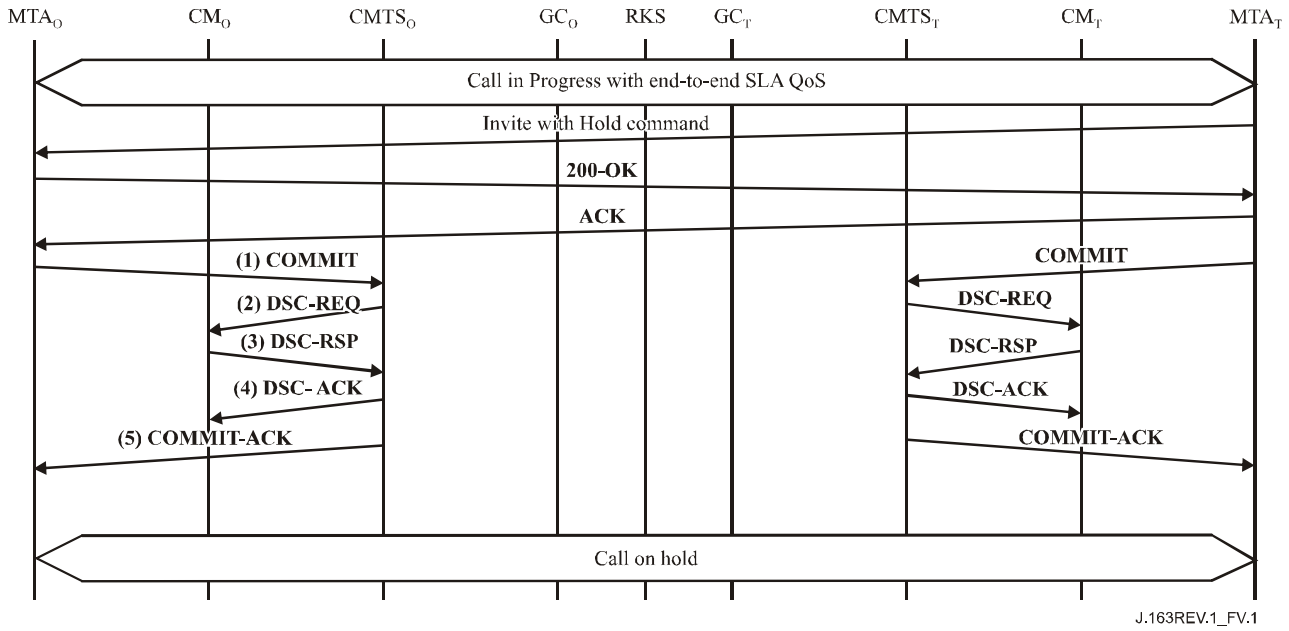


Figure V.1/J.163 – QoS signalling for Call Hold

1) When MTA decides that the current call is to be placed in hold it sends a commit message with a bandwidth of zero. The MTA cannot change the active session ID during a call hold commit message.

COMMIT

Session-Object	Protocol	UDP	The Session-Object and Sender-Template verify the gate identity.
	Destination Address	MTAo	
	Destination port	7120	
Sender Templ	Source Address	MTAt	
	Source port	7000	
Gate-ID		37125	
Flowspec	r	0	These are optional in a COMMIT message, and indicate the activation is for some amount different from the reservation; in this case the desired upstream activation is null.
	b	0	
	p	0	
	m	0	
	M	0	
	R	0	
	S	0	
Reverse-Flowspec	r	0	These are optional in a COMMIT message, and indicate the activation is for some amount different from the reservation; in this case the desired downstream activation is null.
	b	0	
	p	0	
	m	0	
	M	0	
	R	0	
	S	0	

2) The CMTS sends the CM a DSC-REQ message to deactivate the Service Flow and to deactivate the classifiers.

DSC-REQ

TransactionID		1
UpstreamServiceFlow	ServiceFlowIdentifier	1001
	QoSParameterSetType	Admitted (2)
	TimeOutAdmitted	200
	ServiceFlowScheduling	UGS (6)
	Request/TransmissionPolicy	0x00000017
	NominalGrantInterval	10 ms
	ToleratedGrantJitter	2 ms
	GrantsPerInterval	1
	UnsolicitedGrantSize	111
DownstreamServiceFlow	ServiceFlowIdentifier	2001
	QoSParameterSetType	Admitted (2)
	TimeOutAdmitted	200
	TrafficPriority	5
	MaximumSustainedRate	12000

DSC-REQ

UpstreamPacketClassification	ServiceFlowIdentifier	1001
	PacketClassifierIdentifier	3001
	ClassifierPriority	150
	ClassifierActivationState	Inactive (0)
	IPSourceAddress	MTAo
	IPSourcePort	7120
	IPDestinationAddress	MTAt
	IPDestinationPort	7000
	IPProtocol	UDP (17)
DownstreamPacketClassification	ServiceFlowIdentifier	2001
	PacketClassifierIdentifier	3002
	ClassifierPriority	150
	ClassifierActivationState	Inactive (0)
	IPSourceAddress	MTAt
	IPSourcePort	7000
	IPDestinationAddress	MTAo
	IPDestinationPort	7124
	IPProtocol	UDP (17)
HMAC		

3) The CM sends a DSC-RSP message showing the operation was successful.

DSC-RSP

TransactionID		2
ConfirmationCode		Success (0)
HMAC		

4) The CMTS sends DSC-ACK message to indicate that the DSC-RSP has been received and agreed.

DSC-ACK

TransactionID		2
ConfirmationCode		Success (0)
HMAC		

5) The CMTS sends COMMIT-ACK message.

COMMIT-ACK

Session-Object	Protocol	UDP	The Session-Object and Sender-Template verify the gate identity.
	Destination Address	MTAo	
	Destination port	7120	
Sender Templ	Source Address	MTAt	
	Source port	7000	
Gate-ID		37125	

Appendix VI

Sample protocol message exchanges for Call Waiting

VI.1 Example call flow

See Figure VI.1.

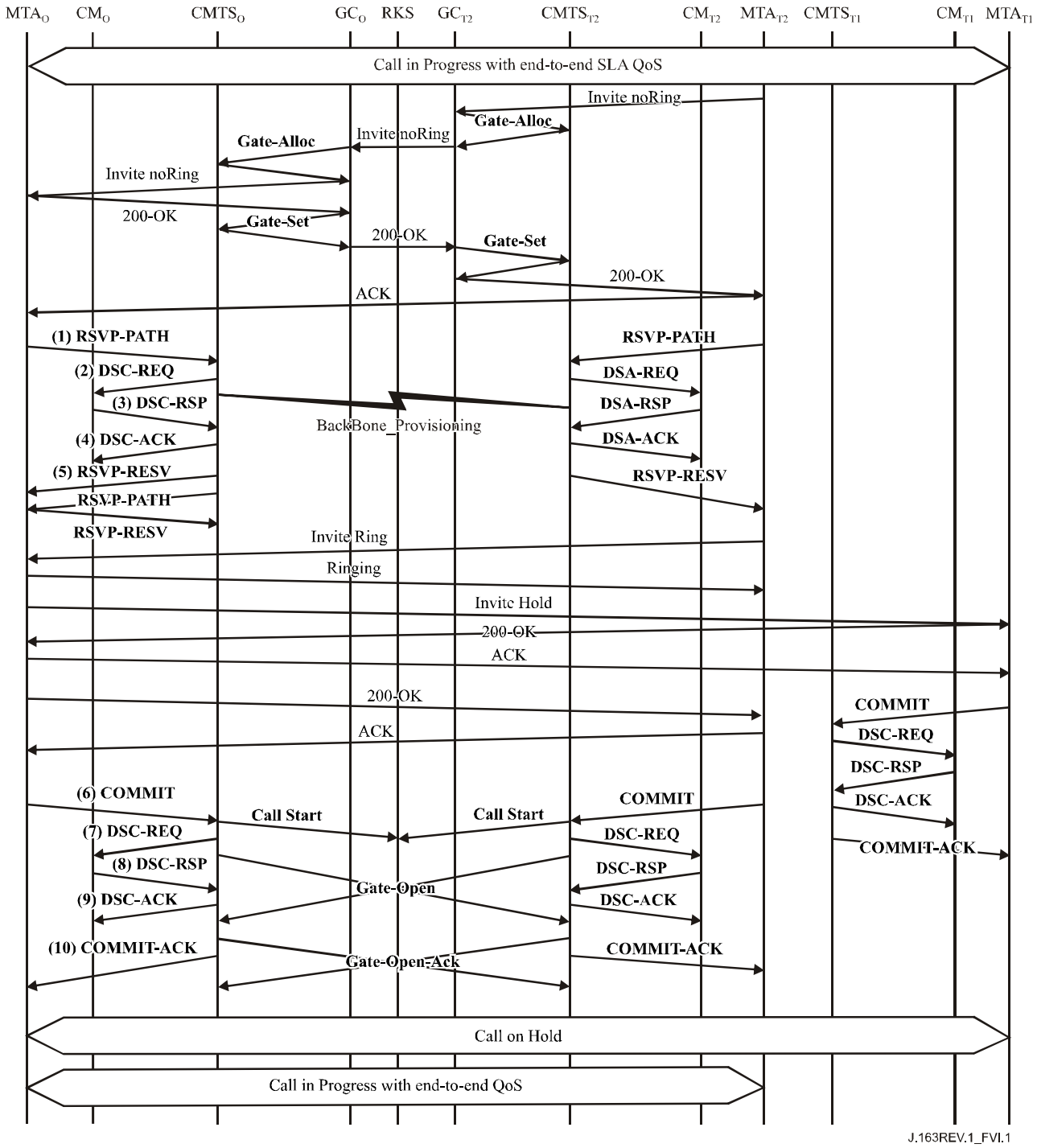


Figure VI.1/J.163 – QoS signalling for Call Waiting

1) MTAo is connected to MTAt1, and receives an incoming call from MTAt2. For this example, assume the call from MTAt1 had been using UDP port 7120, and assigned ResourceID 472. Upon receipt of the call signalling information, MTAo sends an RSVP-PATH message, addressed to MTAt2, but with the Router-Alert bit set in the IP header. Intermediate routers in the home LAN intercept, process, and forward this message as a normal RSVP-PATH, thinking it is a separate flow and allocating separate resources for it.

RSVP-PATH

Session-Object	Protocol	UDP	The parameters form the classifier, matching the authorization previously sent by the GateController.
	Destination Address	MTAt2	
	Destination port	7000	
Sender Templ	Source Address	MTAo	
	Source port	7122	
Sender-Tspec	r	12000	These are the negotiated traffic parameters actually being requested for this call. The CMTS calculates the actual upstream QoS parameters using these Tspec and Rspec parameters. This is a standard RSVP object, which will be interpreted by all intermediate routers in the path between the MTA and CMTS.
	b	120	
	p	12000	
	m	120	
	M	120	
	Hdr Suppression	40	
	VAD	off	
Forward Rspec	R	12000	
	S	0	
Reverse-Session	Protocol	UDP	New RSVP objects that provide the CMTS with sufficient information to calculate downstream traffic parameters and to generate an RSVP-PATH message for the downstream flow.
	Destination Addr	MTAo	
	Destination port	7122	
Reverse-Sender Templ	Source Address	MTAt	
	Source port	0	
Reverse-Sender-Tspec	r	12000	Negotiated traffic parameters actually being requested for this call. The CMTS calculates the actual downstream QoS parameters using these Tspec and Rspec parameters. This is a new RSVP object, which will be ignored by intermediate routers.
	b	120	
	p	12000	
	m	120	
	M	120	
	Hdr Suppression	0	
	VAD	off	
Reverse-Rspec	R	12000	
	S	0	
ResourceID		472	Resource ID assigned for existing call.
Gate-ID		37126	Gate-ID for this new call, take resources from old.

2) The CMTS uses the RSVP-PATH message and calculates the QoS parameters for the J.112 link. For this example, assume the previous call was also G.711, and therefore the bandwidth requirements are identical. Thus the existing ServiceFlow can be used for both packet streams. The CMTS sends the following DSC-REQ to the CM, which establishes the new classifiers. Header suppression, being specified as length 40 in the RSVP-PATH, indicates the 42 bytes of Ethernet/IP/UDP header. Contents of the suppressed header is taken from the RSVP packet.

DSC-REQ

TransactionID		1
UpstreamPacketClassification	ServiceFlowIdentifier	1001
	PacketClassifierIdentifier	3003
	ClassifierChangeAction	Add (0)
	ClassifierPriority	150
	ClassifierActivationState	Inactive (0)
	IPSourceAddress	MTAo
	IPSourcePort	7122
	IPDestinationAddress	MTAt2
	IPDestinationPort	7000
	IPProtocol	UDP (17)
DownstreamPacketClassification	ServiceFlowIdentifier	2001
	PacketClassifierIdentifier	3004
	ClassifierPriority	150
	ClassifierActivationState	Inactive (0)
	IPSourceAddress	MTAt2
	IPDestinationAddress	MTAo
	IPDestinationPort	7122
	IPProtocol	UDP (17)
PayloadHeaderSuppression	ClassifierIdentifier	3003
	ServiceFlowIdentifier	1001
	HeaderSuppressionIndex	1
	HeaderSuppressionField	<42bytes>
	HeaderSuppressionMask	<42bits>
	HeaderSuppressionSize	42
	HeaderSuppressionVerify	Verify (0)
HMAC		

3) The CM checks the resources it is required to allocate (e.g., header suppression table space, Service Flow IDs, classifier table space, local network bandwidth), and installs the classifiers. If the operation is successful, it returns the DSC-RSP message stating the success.

DSC-RSP

TransactionID		1
ConfirmationCode		Success (0)
HMAC		

4) Upon receipt of the DSC-RSP, the CMTS acknowledges receipt with a DSC-ACK message.

DSC-ACK

TransactionID		1
ConfirmationCode		Success (0)
HMAC		

5) Once the J.112 reservation is complete, and the backbone reservation is successful, the CMTS responds to the RSVP-PATH message by sending an RSVP-RESV message. The message includes the ResourceID that is assigned by the CMTS to this connection. The RSVP-RESV message is sent with the source address of MTAt and destination address of MTAo. All intermediate routers will intercept, process, and forward this as a standard RSVP-RESV message.

RSVP-RESV

Session-Object	Protocol	UDP	These fields identify the IP flow for which the reservation is being established.
	Destination Address	MTAt2	
	Destination port	7000	
Filter-Spec	Source Address	MTAo	
	Source port	7122	
Flowspec	r	12000	These fields identify the resources being reserved for this flow.
	b	120	
	p	12000	
	m	120	
	M	120	
	R	12000	
	S	0	
ResourceID		472	Resource ID for this reservation.

6) In response to a hookflash, and after performing further signalling with both the previous and new parties, MTAo sends the COMMIT message to the CMTS. This message is directed to the CMTS at a UDP port determined via call signalling.

COMMIT

Session-Object	Protocol	UDP	The protocol, Destination Address, Source Address, and Destination Port must match those of the Gate ID.
	Destination Address	MTAt2	
	Destination port	7000	
Sender Templ	Source Address	MTAo	
	Source port	7122	
Gate-ID		37126	

7) The Session-Object and Sender-Template give the CMTS enough information to identify the "gate" and to identify which reserved resources are being committed. Since no Tspecs are given in this message, all the reserved resources will be activated. All other flows assigned the same ResourceID will be deactivated.

8) The CMTS resolves which reservation is to be activated, and sends a DSC-REQ to the CM to activate the flow.

DSC-REQ

TransactionID		2
UpstreamServiceFlow	ServiceFlowIdentifier	1001
	QoSParameterSetType	Admitted + Activated (6)
	TimeOutActive	10
	ServiceFlowScheduling	UGS (6)
	NominalGrantInterval	10 ms
	ToleratedGrantJitter	2 ms
	GrantsPerInterval	1
	UnsolicitedGrantSize	111
DownstreamServiceFlow	ServiceFlowIdentifier	2001
	QoSParameterSetType	Admitted + Activated (6)
	TimeOutActive	10
	TrafficPriority	5
	MaximumSustainedRate	12000
Upstream Classifier	ServiceFlowIdentifier	1001
	PacketClassifierIdentifier	3001
	ClassifierChangeAction	Replace (1)
	ClassifierPriority	150
	ClassifierActivationState	Inactive (0)
	IPSourceAddress	MTAo
	IPSourcePort	7120
	IPDestinationAddress	MTAt
	IPDestinationPort	7000
	IPProtocol	UDP (17)
Downstream Classifier	ServiceFlowIdentifier	2001
	PacketClassifierIdentifier	3002
	ClassifierChangeAction	Replace (1)
	ClassifierPriority	150
	ClassifierActivationState	Inactive (0)
	IPSourceAddress	MTAt
	IPSourcePort	7000
	IPDestinationAddress	MTAo
	IPDestinationPort	7120
	IPProtocol	UDP (17)

DSC-REQ

UpstreamPacketClassification	ServiceFlowIdentifier	1001
	PacketClassifierIdentifier	3003
	ClassifierChangeAction	Replace (1)
	ClassifierPriority	150
	ClassifierActivationState	Active (1)
	IPSourceAddress	MTAo
	IPSourcePort	7122
	IPDestinationAddress	MTAt2
	IPDestinationPort	7000
	IPProtocol	UDP (17)
DownstreamPacketClassification	ServiceFlowIdentifier	2001
	PacketClassifierIdentifier	3004
	ClassifierChangeAction	Replace (1)
	ClassifierPriority	150
	ClassifierActivationState	Active (1)
	IPSourceAddress	MTAt2
	IPSourcePort	7000
	IPDestinationAddress	MTAo
	IPDestinationPort	7122
	IPProtocol	UDP (17)
HMAC		

9) The CM sends a DSC-RSP message showing the operation was successful.

DSC-RSP

TransactionID		2
ConfirmationCode		Success (0)
HMAC		

10) The CMTS sends DSC-ACK message to indicate that the DSC-RSP has been received and agreed.

DSC-ACK

TransactionID		2
ConfirmationCode		Success (0)
HMAC		

11) The CMTS acknowledges the COMMIT with:

COMMIT-ACK

Session-Object	Protocol	UDP	The protocol, Destination Address, Source Address, and Destination Port quadruple matches Gate ID.
	Destination Address	MTAt2	
	Destination port	7000	
Sender Templ	Source Address	MTAo	
	Source port	7122	
Gate-ID		37126	

Appendix VII

Sample protocol message exchanges for basic DCS on-net to on-net call of an embedded MTA

See Figure VII.1.

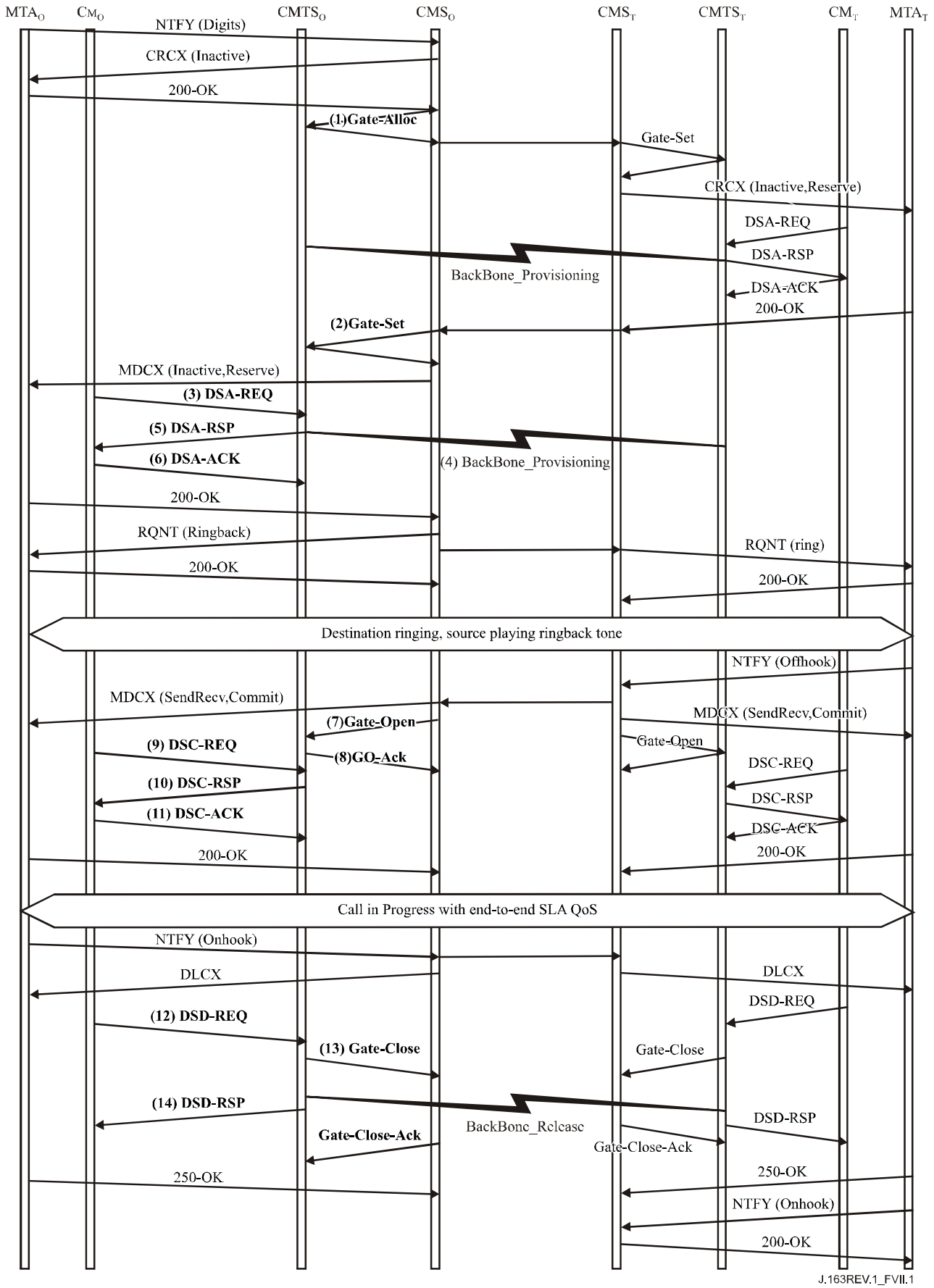


Figure VII.1/J.163 – Basic call flow – Embedded MTA

1) CMSo, upon receipt of signalling information from MTAo, checks the current resource consumption of MTAo by consulting CMTSo.

GATE-ALLOC

TransactionID		3176	
Subscriber		MTAo	Request for total resources in use by this client.
Activity-Count		12	Maximum connections allowed by client.

CMTSo checks current resource usage by MTAo, and responds telling the number of connections active.

GATE-ALLOC-ACK

TransactionID		3176	
Subscriber		MTAo	Request for total resources in use by this client.
Gate-ID		37125	Identifier for allocated Gate.
Activity Count		3	Total connections established by this client.

2) CMSo, upon further signalling exchanges, gives CMTSo authorization to admit the new connection.

GATE-SET

Transaction ID		3177	Unique Transaction ID for this message exchange.
Subscriber		MTAo	Request for total resources in use by this client.
Gate-ID		37125	Identifier for allocated Gate.
Remote-Gate-Info	CMTS Address	CMTSo	Information needed to perform gate coordination. Note that CMS has given itself as the entity for exchanging gate coordination messages.
	CMTS Port	2052	
	Remote Gate-ID	8095	
	Security Key	<key>	
Event-Generation-Info	RKS-Addr	RKS	Address of Record Keeping Server.
	RKS-Port	3288	Port on Record Keeping Server.
	Billing Correlation ID	<id>	Opaque data that will be passed to RKS when resources are committed.

GATE-SET

Gate-Spec	Direction	up	
	Protocol	UDP	The protocol, Destination Address, Source Address, and Destination Port quadruple are used for QoS classifiers.
	Source Address	MTAo	
	Destination Address	MTAt	
	Source port	0	
	Destination port	7000	
	DSCP	6	Packet Type value for upstream packets.
	T1	180000	Maximum time between reserve and commit.
	T2	2000	Maximum time for gate coordination to complete.
	r	12000	These are the maximum bandwidth parameters that MTAo is authorized to request for this conversation.
	b	120	
	p	12000	
	m	120	
	M	120	
	R	12000	
	S	0	
Gate-Spec	Direction	down	
	Protocol	UDP	The protocol, Destination Address, Source Address, and Destination Port quadruple are used for QoS classifiers.
	Source Address	MTAt	
	Destination Address	MTAo	
	Source port	0	
	Destination port	7120	
	DSCP	9	Packet Type value for downstream packets.
	T1	180000	Maximum time between reserve and commit.
	T2	2000	Maximum time for gate coordination to complete.
	r	12000	These are the maximum bandwidth parameters that MTAo is authorized to request for this conversation.
	b	120	
	p	12000	
	m	120	
	M	120	
	R	12000	
	S	0	

CMTSo responds to the Gate Set-up command with an acknowledgement.

GATE-SET-ACK

TransactionID		3177	
Subscriber		MTAo	Request for total resources in use by this client.
Gate-ID		37125	Identifier for allocated Gate.
Activity Count		4	Total connections established by this client.

3) MTAo, upon receiving call signalling information, calculates the QoS parameters for the J.112 link. It uses the Annex E to Annex B/J.112 interface to the CM to send the following DSA-REQ to the CMTS. This message is used to establish both upstream and downstream parameters. The Upstream Unsolicited Grant Size was calculated as 120 (from SDP) plus 18 (Ethernet overhead) minus 40 (Header Suppression amount) plus 13 (J.112 overhead). Header suppression indicates the 42 bytes of Ethernet/IP/UDP header. Contents of the suppressed header is included in the DSA-REQ.

DSA-REQ

TransactionID		1
UpstreamServiceFlow	ServiceFlowReference	1
	QoSParameterSetType	Admitted (2)
	TimeOutAdmitted	200
	ServiceFlowScheduling	UGS (6)
	NominalGrantInterval	10 ms
	ToleratedGrantJitter	2 ms
	GrantsPerInterval	1
	UnsolicitedGrantSize	111
DownstreamServiceFlow	ServiceFlowReference	2
	QoSParameterSetType	Admitted (2)
	TimeOutAdmitted	200
	TrafficPriority	5
	MaximumSustainedRate	12000
UpstreamPacketClassification	ServiceFlowReference	1
	PacketClassifierReference	1
	ClassifierPriority	150
	ClassifierActivationState	Inactive (0)
	IPSourceAddress	MTAo
	IPSourcePort	7120
	IPDestinationAddress	MGt
	IPDestinationPort	7000
	IPProtocol	UDP (17)

DSA-REQ

DownstreamPacketClassification	ServiceFlowReference	2
	PacketClassifierReference	2
	ClassifierPriority	150
	ClassifierActivationState	Inactive (0)
	IPSourceAddress	MGt
	IPDestinationAddress	MTAo
	IPDestinationPort	7124
	IPProtocol	UDP (17)
PayloadHeaderSuppression	ClassifierReference	1
	ServiceFlowReference	1
	HeaderSuppressionIndex	1
	HeaderSuppressionField	<42bytes>
	HeaderSuppressionMask	<42bits>
	HeaderSuppressionSize	42
	HeaderSuppressionVerify	Verify (0)
AuthorizationBlock		37125
HMAC		

4) Simultaneous with message No. 5, the CMTS initiates any required backbone reservations for the requested quality of service. The content of this message is dependent on the particular backbone algorithms in use, and is outside the scope of this Recommendation. The backbone router sends to the CMTS any required notification that the reservation is successful.

5) The CMTS checks the authorization, by looking for a gate with Gate-ID matching the value in AuthBlock, and checks the resources it is required to allocate (e.g., header suppression table space, Service Flow IDs, classifier table space), and installs the classifiers. If the operation is successful, it returns the DSA-RSP message stating the success.

DSA-RSP

TransactionID		1
ConfirmationCode		Success (0)
UpstreamServiceFlow	ServiceFlowReference	1
	ServiceFlowIdentifier	1001
	QoSParameterSetType	Admitted (2)
	TimeOutAdmitted	200
	ServiceFlowScheduling	UGS (6)
	NominalGrantInterval	10 ms
	ToleratedGrantJitter	2 ms
	GrantsPerInterval	1
	UnsolicitedGrantSize	111

DSA-RSP

DownstreamServiceFlow	ServiceFlowReference	2
	ServiceFlowIdentifier	2001
	QoSParameterSetType	Admitted (2)
	TimeOutAdmitted	200
	TrafficPriority	5
	MaximumSustainedRate	12000
UpstreamPacketClassification	ServiceFlowReference	1
	PacketClassifierReference	1
	PacketClassifierIdentifier	3001
	ClassifierPriority	150
	ClassifierActivationState	Inactive (0)
	IPSourceAddress	MTAo
	IPSourcePort	7120
	IPDestinationAddress	MGt
	IPDestinationPort	7000
	IPProtocol	UDP (17)
DownstreamPacketClassification	ServiceFlowReference	2
	PacketClassifierReference	2
	PacketClassifierIdentifier	3002
	ClassifierPriority	150
	ClassifierActivationState	Active (1)
	IPSourceAddress	MGt
	IPDestinationAddress	MTAo
	IPDestinationPort	7124
	IPProtocol	UDP (17)
HMAC		

6) Upon receipt of the DSA-RSP, the CM acknowledges receipt with a DSA-ACK message.

DSA-ACK

TransactionID		1
ConfirmationCode		Success (0)
HMAC		

7) The CMS sends the gate open message to the CMTS to inform it that the resources should be committed. If the CMTS does not receive the DSC-REQ from MTAo within a short time, it should revoke the gate authorization.

GATE-OPEN

Transaction ID		72	Identifier to match this message with its response.
Gate ID		37125	Gate-ID at CMTS.
HMAC			Security checksum for this message.

8) The CMTS responds to the GATE-OPEN with:

GATE-OPEN-ACK

Transaction ID		72	Identifier to match this message with its response.
HMAC			Security checksum for this message.

9) In response to signalling messages that indicate the call has completed (i.e., the other side has gone off-hook), MTAo uses the Annex E to Annex B/J.112 interface to activate the admitted resources. This is done via a DSC-REQ command to the CMTS.

DSC-REQ

TransactionID		2
UpstreamServiceFlow	ServiceFlowIdentifier	1001
	QoSParameterSetType	Admitted + Activated (6)
	TimeOutActive	10
	ServiceFlowScheduling	UGS (6)
	NominalGrantInterval	10 ms
	ToleratedGrantJitter	2 ms
	GrantsPerInterval	1
	UnsolicitedGrantSize	111
DownstreamServiceFlow	ServiceFlowIdentifier	2001
	QoSParameterSetType	Admitted + Activated (6)
	TimeOutActive	10
	TrafficPriority	5
	MaximumSustainedRate	12000
UpstreamPacketClassification	ServiceFlowIdentifier	1001
	PacketClassifierIdentifier	3001
	ClassifierChangeAction	Replace (1)
	ClassifierPriority	150
	ClassifierActivationState	Active (1)
	IPSourceAddress	MTAo
	IPSourcePort	7120
	IPDestinationAddress	MGt
	IPDestinationPort	7000
	IPProtocol	UDP (17)

DSC-REQ

DownstreamPacketClassification	ServiceFlowIdentifier	2001
	PacketClassifierIdentifier	3002
	ClassifierChangeAction	Replace (1)
	ClassifierPriority	150
	ClassifierActivationState	Active (1)
	IPSourceAddress	MGt
	IPSourcePort	7000
	IPDestinationAddress	MTAo
	IPDestinationPort	7124
	IPProtocol	UDP (17)
HMAC		

10) The CMTS sends a DSC-RSP message showing the operation was successful.

DSC-RSP

TransactionID		2
ConfirmationCode		Success (0)
HMAC		

11) The CM sends a DSC-ACK message to indicate that the DSC-RSP has been received and agreed.

DSC-ACK

TransactionID		2
ConfirmationCode		Success (0)
HMAC		

12) When the call is finished the MTA uses the Annex E to Annex B/J.112 interface to delete the Service Flows, sending a DSD-REQ message to the CMTS.

DSD-REQ

TransactionID		3
ServiceFlowID		1001
HMAC		

DSD-REQ

TransactionID		4
ServiceFlowID		2001
HMAC		

13) The CMTS, upon receiving DSD-REQ, sends the gate coordination message to the CMS (identified in the Gate-Set message).

GATE-CLOSE

Transaction ID		73	Identifier to match this message with its response.
Gate-ID		8095	This identifies the GateID at the CMS.
HMAC			Security checksum for this message.

The remote CMTS responds with:

GATE-CLOSE-ACK

Transaction ID		73	Identifier to match this message with its response.
HMAC			Security checksum for this message.

14) The CMTS deletes the Service Flow IDs and sends the response to the CM.

DSD-RSP

TransactionID		3
ServiceFlowID		1001
ConfirmationCode		Success (0)
HMAC		

DSD-RSP

TransactionID		4
ServiceFlowID		2001
ConfirmationCode		Success (0)
HMAC		

Appendix VIII

Sample protocol message exchanges for basic NCS call for embedded MTA

See Figure VIII.1.

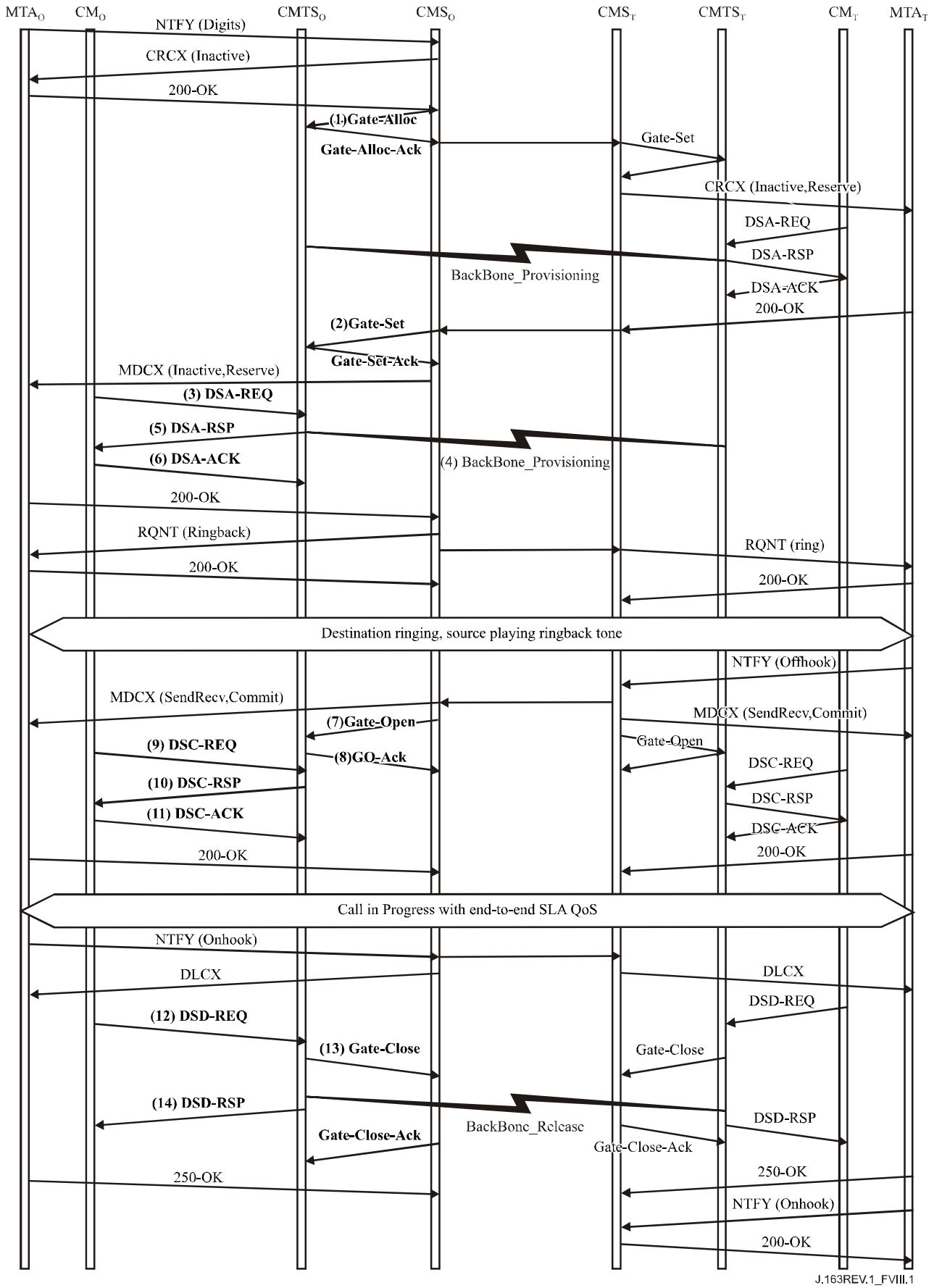


Figure VIII.1/J.163 – On-net to on-net embedded NCS call

1) CMSo, upon receipt of signalling information from MTAo, checks the current resource consumption of MTAo by consulting CMTSo.

GATE-ALLOC

TransactionID		3176	
Subscriber		MTAo	Request for total resources in use by this client.
Activity-Count		12	Maximum gates allowed by client.

CMTSo checks current resource usage by MTAo, and responds telling the number of connections active.

GATE-ALLOC-ACK

TransactionID		3176	
Subscriber		MTAo	Request for total resources in use by this client.
Gate-ID		37125	Identifier for allocated Gate.
Activity Count		3	Total connections established by this client.
Gate Coordination port		7890	UDP port on which CMTSo will listen for gate coordination messages for this gate.

2) CMSo, upon further signalling exchanges, gives CMTSo authorization to admit the new connection.

GATE-SET

TransactionID		3177	Unique Transaction ID for this message exchange.
Subscriber		MTAo	Request for total resources in use by this client.
Gate-ID		37125	Identifier for allocated Gate.
Remote-Gate-Info	CMTS Address	CMSo	Information needed to perform gate coordination. Note that CMS has given itself as the entity for exchanging gate coordination messages.
	CMTS Port	2052	
	Remote Gate-ID	8095	
	Security Key	<key> (16 octets)	
	Flag	2 (No-Send-Gate-Open)	
	Authentication Algorithm	100 (MD5MAC)	

GATE-SET

Event-Generation-Info	RKS-Addr-1	RKS-1	Address of Primary Record Keeping Server.
	RKS-Port-1	3288	Port on Primary Record Keeping Server.
	Flags	0	Do not batch.
	RKS-Addr-2	RKS-2	Address of secondary Record Keeping Server.
	RKS-Port-2	3288	Address of secondary Record Keeping Server.
	Billing Correlation ID	<id>	Opaque data that will be passed to RKS when resources are committed.
Gate-Spec	Direction	1 (up)	
	Protocol	17 (UDP)	The protocol, Destination Address, Source Address, and Destination Port quadruple are used.
	Flags	0	
	Session Class	1 (normal priority)	
	Source Address	MTAo	
	Destination Address	MTAt	
	Source port	0	
	Destination port	7000	
	DS	5	DiffServ Code Point value for upstream packets.
	T1	300	Maximum time between reserve and commit.
	T2	2	Maximum time for gate coordination to complete.
	b	120	These are the maximum bandwidth parameters that MTAo is authorized to request for this conversation.
	r	12000	
	p	12000	
	m	120	
	M	120	
	R	12000	
	S	800	

GATE-SET

Gate-Spec	Direction	0 (down)	
	Flag	0	
	Protocol	17 (UDP)	The protocol, Destination Address, Source Address, and Destination Port quadruple are used for QoS classifiers.
	Session Class		
	Source Address	MTAt	
	Destination Address	MTAo	
	Source port	0	
	Destination port	7120	
	DS	5	Packet Type value for downstream packets.
	T1	300	Maximum time between reserve and commit.
	T2	2	Maximum time for gate coordination to complete.
	b	120	These are the maximum bandwidth parameters that MTAo is authorized to request for this conversation.
	r	12000	
	p	12000	
	m	120	
	M	120	
	R	12000	
	S	0	

CMTSo responds to the Gate Set-up command with an acknowledgement.

GATE-SET-ACK

TransactionID		3177	
Subscriber		MTAo	
Gate-ID		37125	Identifier for allocated Gate.
Activity Count		4	Total connections established by this client.
Gate Coordination Port		7890	UDP port on which CMTSo will listen for gate coordination messages for this gate Required for the second gate (which was created by the gate set) Both gates share the same ID.

3) MTAo, upon receiving call signalling information, calculates the QoS parameters for the J.112 link. It uses the Annex E to Annex B/J.112 interface to the CM to send the following DSA-REQ to the CMTS. This message is used to establish both upstream and downstream parameters. The Upstream Unsolicited Grant Size was calculated as 120 (from SDP) plus 18 (Ethernet overhead) plus 14 (DOCSIS overhead).

DSA-REQ

TransactionID		1
UpstreamServiceFlow	ServiceFlowReference	1
	QoSParameterSetType	Admitted (2)
	ServiceFlowScheduling	UGS (6)
	Request/Transmission Policy	0x0000017F
	NominalGrantInterval	10 ms
	ToleratedGrantJitter	800 μs
	GrantsPerInterval	1
	UnsolicitedGrantSize	152
DownstreamServiceFlow	ServiceFlowReference	2
	QoSParameterSetType	Admitted (2)
	MinimumReservedRateRate	110400
	AssumedMinResRatePktSiz	138
UpstreamPacketClassification	ServiceFlowReference	1
	PacketClassifierReference	1
	ClassifierPriority	128
	ClassifierActivationState	Inactive (0)
	IPSourceAddress	MTAo
	IPSourcePortStart	7120
	IPSoursePortEnd	7120
	IPDestinationAddress	MTAt
	IPDestinationPortStart	7000
	IPDestinationPortEnd	7000
	IPProtocol	UDP (17)
DownstreamPacketClassification	ServiceFlowReference	2
	PacketClassifierReference	2
	ClassifierPriority	128
	ClassifierActivationState	Inactive (0)
	IPSourceAddress	MTAt
	IPDestinationAddress	MTAo
	IPDestinationPortStart	7120
	IPDestinationPortEnd	7120
	IPProtocol	UDP (17)
AuthorizationBlock		37125
HMAC		

4) Simultaneous with message number 5, the CMTS initiates any required backbone reservations for the requested quality of service. The content of this message is dependent on the particular backbone algorithms in use, and is outside the scope of this Recommendation. The backbone router sends to the CMTS any required notification that the reservation is successful.

5) The CMTS checks the authorization, by looking for a gate with gate-ID matching the value in AuthBlock, and checks the resources it is required to allocate (e.g., header suppression table space, Service Flow IDs, classifier table space), and installs the classifiers. If the operation is successful, it returns the DSA-RSP message stating the success.

DSA-RSP

TransactionID		1
ConfirmationCode		Success (0)
UpstreamServiceFlow	ServiceFlowReference	1
	ServiceFlowIdentifier	1001
	ServiceID	801
	QoSParameterSetType	Admitted (2)
	TimeOutAdmitted	200
	ServiceFlowScheduling	UGS (6)
	Request/TransPolicy	0x0000017F
	NominalGrantInterval	10 ms
	ToleratedGrantJitter	800 μs
	GrantsPerInterval	1
	UnsolicitedGrantSize	152
DownstreamServiceFlow	ServiceFlowReference	2
	ServiceFlowIdentifier	2001
	QoSParameterSetType	Admitted (2)
	MinimumReservedRate	110400
	AssumedMinResRatePktSiz	138
UpstreamPacketClassification	ServiceFlowReference	1
	ServiceFlowIdentifier	1001
	PacketClassifierReference	1
	PacketClassifierIdentifier	3001
	ClassifierPriority	128
	ClassifierActivationState	Inactive (0)
	IPSourceAddress	MTAo
	IPSourcePortStart	7120
	IPSourcePortEnd	7120
	IPDestinationAddress	MTAt
	IPDestinationPortStart	7000
	IPDestinationPortEnd	7000
	IPProtocol	UDP (17)

DSA-RSP

DownstreamPacketClassification	ServiceFlowReference	2
	ServiceFlowIdentifier	2001
	PacketClassifierReference	2
	PacketClassifierIdentifier	3002
	ClassifierPriority	128
	ClassifierActivationState	Inactive (0)
	IPSourceAddress	MTAt
	IPDestinationAddress	MTAo
	IPDestinationPortStart	7120
	IPDestinationPortEnd	7120
	IPProtocol	UDP (17)
Authorization Block	Gate-ID	37125
	Resource ID	71209
HMAC		

6) Upon receipt of the DSA-RSP, the CM acknowledges receipt with a DSA-ACK message.

DSA-ACK

TransactionID		1
ConfirmationCode		Success (0)
HMAC		

7) The CMS sends the gate open message to the CMTS to inform it that the resources should be committed. If the CMTS does not receive the DSC-REQ from MTAo within a short time, it should revoke the gate authorization.

GATE-OPEN

TransactionID		72	Identifier to match this message with its response.
Gate ID		37125	Gate-ID at CMTS.

8) The CMTS responds to the GATE-OPEN with:

GATE-OPEN-ACK

TransactionID		72	Identifier to match this message with its response.
GateID		37125	GATE-ID at CMTS.

9) In response to signalling messages that indicate the call has completed (i.e., the other side has gone off-hook), MTAo uses the interface to activate the admitted resources. This is done via a DSC-REQ command to the CMTS.

DSC-REQ

TransactionID		2
UpstreamServiceFlow	ServiceFlowIdentifier	1001
	QoSParameterSetType	Admitted + Activated (6)
	ServiceFlowScheduling	UGS (6)
	Request/TransPolicy	0x0000017F
	NominalGrantInterval	10 ms
	ToleratedGrantJitter	800 μs
	GrantsPerInterval	1
	UnsolicitedGrantSize	152
DownstreamServiceFlow	ServiceFlowIdentifier	2001
	QoSParameterSetType	Admitted + Activated (6)
	MinimumReservedRate	110400
	AssumedMinResRatePktSiz	138
UpstreamPacketClassification	ServiceFlowIdentifier	1001
	PacketClassifierIdentifier	3001
	ClassifierChangeAction	Replace (1)
	ClassifierPriority	128
	ClassifierActivationState	Active (1)
	IPSourceAddress	MTAo
	IPSourcePortStart	7120
	IPSourcePortEnd	7120
	IPDestinationAddress	MTAt
	IPDestinationPortStart	7000
	IPDestinationPortEnd	7000
	IPProtocol	UDP (17)
DownstreamPacketClassification	ServiceFlowIdentifier	2001
	PacketClassifierIdentifier	3002
	ClassifierChangeAction	Replace (1)
	ClassifierPriority	128
	ClassifierActivationState	Active (1)
	IPSourceAddress	MTAt
	IPDestinationAddress	MTAo
	IPDestinationPortStart	7120
	IPDestinationPortEnd	7120
	IPProtocol	UDP (17)
Authorization Block	Gate ID	37125
HMAC		

10) The CMTS sends a DSC-RSP message showing the operation was successful.

DSC-RSP

TransactionID		2
ConfirmationCode		Success (0)
UpstreamServiceFlow	ServiceFlowIdentifier	1001
	ServiceID	801
	QoSParameterSetType	Admitted + Activated (6)
	TimeOutAdmitted	200 s
	TimeOutActive	10 s
	ServiceFlowScheduling	UGS (6)
	Request/TransPolicy	0x0000017F
	NominalGrantInterval	10 ms
	ToleratedGrantJitter	800 µs
	GrantsPerInterval	1
	UnsolicitedGrantSize	152
DownstreamServiceFlow	ServiceFlowIdentifier	2001
	QoSParameterSetType	Admitted + Activated (6)
	MinimumReservedRate	110400
	AssumedMinResRatePktSiz	138
UpstreamPacketClassification	ServiceFlowIdentifier	1001
	PacketClassifierIdentifier	3001
	ClassifierChangeAction	Replace (1)
	ClassifierPriority	128
	ClassifierActivationState	Active (1)
	IPSourceAddress	MTAo
	IPSourcePortStart	7120
	IPSourcePortEnd	7120
	IPDestinationAddress	MTAt
	IPDestinationPortStart	7000
	IPDestinationPortEnd	7000
	IPProtocol	UDP (17)
DownstreamPacketClassification	ServiceFlowIdentifier	2001
	PacketClassifierIdentifier	3002
	ClassifierChangeAction	Replace (1)
	ClassifierPriority	128
	ClassifierActivationState	Active (1)
	IPSourceAddress	MTAt
	IPDestinationAddress	MTAo
	IPDestinationPortStart	7120
	IPDestinationPortEnd	7120
	IPProtocol	UDP (17)
HMAC		

11) The CM sends DSC-ACK message to indicate that the DSC-RSP has been received and agreed.

DSC-ACK

TransactionID		2
ConfirmationCode		Success (0)
HMAC		

12) When the call is finished the MTA uses the Annex E to Annex B/J.112 interface to delete the Service Flows, sending a DSD-REQ message to the CMTS.

DSD-REQ

TransactionID		3
ServiceFlowID		1001
ServiceFlowID		2001
HMAC		

13) The CMTS, upon receiving DSD-REQ, sends the gate coordination message to the CMS (identified in the Gate-Set message).

GATE-CLOSE

TransactionID		73	Identifier to match this message with its response.
Gate-ID		8095	This identifies the Gate-ID at the CMS.

The CMS responds with:

GATE-CLOSE-ACK

Transaction ID		73	Identifier to match this message with its response.
GateID		8095	Identifies the GateID at the CMS.

14) The CMTS deletes the Service Flow IDs and sends the response to the CM.

DSD-RSP

TransactionID		3
ConfirmationCode		Success (0)
HMAC		

Appendix IX

Theft of service scenarios

We outline here several possible theft of service scenarios to highlight the need for the dynamic authorization, the need for the 2-phase resource reservation protocol, the need for gates, and the need for gate coordination. The system design places much of the session control intelligence at the clients, where it can easily scale with technology and provide new and innovative services. While this "future-proofing" is a goal of the design, we must recognize that it leaves open a wide range of fraud possibilities. This appendix discusses some of those possibilities, and how the QoS signalling architecture prevents them.

The basic assumption is that the MTA is not immune to customer tampering, and that the significant incentive for free service will lead to some very sophisticated attempts to thwart any network controls placed on the MTA. This customer tampering includes, but is not limited to, opening the box and replacing ROMs, replacing integrated circuit chips, probing and reverse engineering of the MTA design, and even total replacement of the MTA with a special black-market version. While technical solutions exist to the physical security of the MTA (e.g., booby trapping the box with lethal gas), they are not considered acceptable.

Since the MTA can be distinguished only by its communication over a J.112 network, it is possible, and quite likely, that PC software will be written that will emulate the behaviour of a MTA. Such a PC may be indistinguishable from a real MTA. The software behaviour in this case is under the total control of the customer.

Further, it is intended that new services will be implemented in the MTA, and that software control of those new services will be provided by a variety of vendors. This updated software will be downloaded into the MTA, leaving open the possibility of customers downloading special hacked versions that provide free service. We do not concern ourselves here with the problem of "trojan horses" in such downloaded software, as this is considered identical to the problem today of customers giving away their credit card numbers and/or PINs. We are concerned with the customer intentionally downloading special software that does only what is in his/her best interest.

IX.1 Scenario No. 1: Customers establishing high QoS Connections themselves

The MTA, with sufficient intelligence, can remember past destinations dialled and the destination address, or use some other mechanism to determine the IP address of a destination. It can then signal that destination itself (with some cooperation of the other client), and negotiate a high quality-of-service connection via the RSVP mechanism or via the Annex E to Annex B/J.112 interface for an embedded client. Since no network agent is used in initiating the session, there will be no billing record produced. Prevention of this scenario is done by requiring dynamic authorization at the CMTS; without the authorization the attempt to obtain the high quality-of-service will fail.

The above scenario required the cooperation of two altered MTAs. Similar theft of service could be accomplished with only the originator being modified. If the originating MTA used the network agent to establish the session, thereby informing the destination in the standard manner of an incoming session, but again negotiated the high quality-of-service itself, there would be no billing record generated and the originator would get a free session. Again, the solution is to require the use of gates in the CMTSs.

IX.2 Scenario No. 2: Customers using provisioned QoS for non-voice applications

Statistically provisioned QoS can only identify a customer as one who is authorized for high Quality of Service. There is no restriction on the usage of the service. In particular, a customer who has subscribed for a commercial-grade voice communications service, and is therefore authorized to activate high-bandwidth low-latency connections through the J.112 network, can use this ability for web surfing or other PC applications. Prevention of this scenario is done by requiring dynamic authorization at the CMTS; without the authorization the attempt to obtain the high quality-of-service will fail.

IX.3 Scenario No. 3: MTA non-cooperation for billing

One can easily imagine what would happen if there was a message from the MTA on session establishment that said, "OK, callee has answered, start billing me now," or a message on hangup that said, "session has completed, stop billing now." However, there are more subtle ways that a user could have the same affect as tinkering with such messages if they existed.

It is essential in providing a commercial-grade voice communications service using IPCablecom to ensure network capacity exists prior to signalling the CPE at the receiving party's location. This function is done with the RESERVE messages. If the RESERVE message were to actually allocate the bandwidth (i.e., combining the RESERVE and COMMIT mechanisms), then there would be no incentive for the MTA to ever issue the COMMIT. The MTA could merely start transmitting voice packets immediately, and the destination could start transmitting voice packets as soon as the phone is answered. The COMMIT message becomes, in effect, the billing start message above. It is therefore essential that the RESERVE not actually allocate the bandwidth, but rather it must check all current allocations and pending reservations to ensure that the bandwidth will be available at the time of a COMMIT message.

IX.4 Scenario No. 4: MTA altering the destination address in voice packets

Another example is when two MTAs, which are far apart, each establish a local session. Once the bandwidth and connection are established, the MTAs then change the IP addresses in the RTP streams to point to each other. The billing system continues to bill each of them for a local session, while the customers are actually engaged in a long distance session. This requires us to have mechanisms at the CMTSs that provide access to higher QoS only based on packet filters previously authorized. Thus, in addition to the 2-phase resource management, this scenario motivates the need for packet filters at the gates.

IX.5 Scenario No. 5: Use of half-connections

This is an example of theft of service that could occur in the absence of gate coordination. Suppose one client in a session sends a COMMIT message and the other does not. For example, say the terminating client sends a COMMIT, but fails to send the proper signalling message, so the originator never sends a COMMIT. In this case, only one gate is opened, and the users and network are left with a half-connection. Given that the originator did not send a COMMIT message, the network cannot legitimately bill the user for the half-connection. However, it is possible for two colluding clients to set up two half-connections, neither of which is billable, which can be combined to give a full connection between the two parties. This results in a free session. Fraud of this type can only be prevented by synchronizing the operation of the two gates.

IX.6 Scenario No. 6: Early termination leaving a half-connection

Gate coordination is also required on completion of the session. Suppose that MTA_O calls MTA_T and pays for the session. Since MTA_O is being charged for the session, it clearly has an incentive to issue a RELEASE message to $CMTS_O$ to close its gate and stop the billing. However, if MTA_T does not issue the RELEASE message to close the gate at $CMTS_T$, a half-connection remains. In this case MTA_T can continue to send voice and/or data to MTA_O without billing for the session. Hence, a GATE-CLOSE message must be issued from the originating side gate at $CMTS_O$ to close the terminating side gate at $CMTS_T$.

IX.7 Scenario No. 7: Forged Gate Coordination messages

Each MTA knows the identity of its CMTS, and knows the 5-tuple that its CMTS uses to identify the GateID. MTAs can do various kinds of end-to-end negotiation before asking for resources; in particular they can easily exchange the information about their GateID. Then the MTA can fake the GATE-OPEN message being sent to the non-paying end, and get a non-billed one-way connection. Doing this twice gets a full non-billed connection. One solution to this problem is for the GateController to give the CMTS a key to use for the CMTS-CMTS messages, on a per-session (or per-gate) basis.

IX.8 Scenario No. 8: Fraud directed against unwanted callers

Due to details of the call set-up sequence, it is possible that the bandwidth authorization at the destination will be more generous than that at the source. Given this, it is then possible for a called party to reserve and allocate bandwidth far in excess of the final negotiated amount, resulting in the calling party being charged for more than expected. If available, this would likely be used against telemarketers, fighting back for unwanted calls during dinner.

Gate coordination, which was used previously to guard against half-connections, also protects from this type of fraud. The GATE-OPEN message tells the bandwidth that was allocated as a result of the COMMIT, and the COMMIT-ACK sent to the originator tells exactly what bandwidth will be charged for the session. If the originator detects anything amiss, he can immediately terminate the session.

Appendix X

COPS (Common Open Policy Service)

X.1 COPS procedures and principles

This appendix provides a brief description of COPS procedures and principles, and how COPS relates to other protocols such as LDAP. COPS is currently defined in an Internet Draft RAP-COPS-07.

Common Open Policy Service (COPS) protocol is a client/server protocol being defined in the IETF RSVP admission policy (rap) working group for use in admission control in RSVP/IntServ and DiffServ QoS networks. COPS runs over TCP/IP, using a well-known port number 3288. COPS entities would reside at a network edge device and a policy server. Three functional entities are defined for the rap framework:

- Policy Decision Point (PDP) – The server entity in COPS, which makes the final decision on session admission or rejection, based on policy information that it has access to. This is expected to be implemented as an application on a stand-alone server device.
- Policy Enforcement Point (PEP) – The client entity in COPS, which consults with the PDP to make policy decisions or to obtain policy information that it may itself use to make admission control decisions; the PEP may receive requests for service and initiate a query to the PDP that will result in a go/no-go response, or the PEP may inform the PDP that it wishes to receive decisions and policy related information on an unsolicited basis.
- Local Decision Point (LDP) – A local version of the PDP that can make decisions based on local information or information cached from previous decisions. A PDP decision always takes precedence over the LDP.

A COPS sequence, as used in an RSVP/IntServ environment, is shown in Figure X.1.

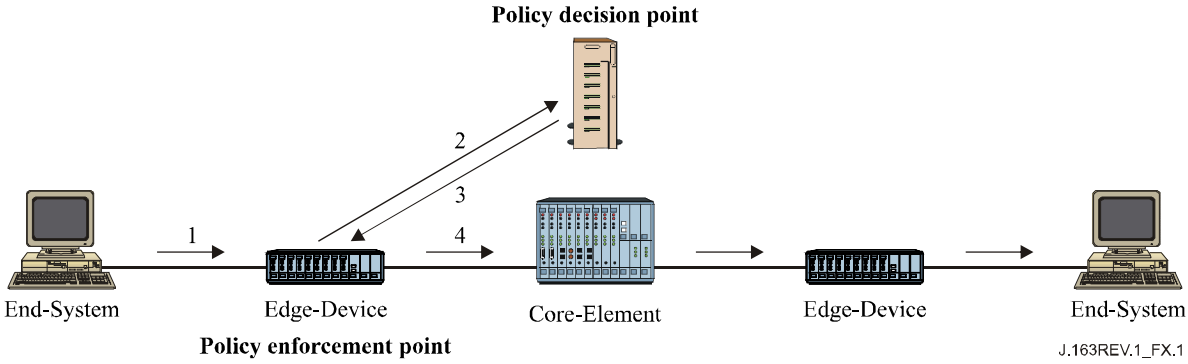


Figure X.1/J.163 – COPS protocol

In the COPS sequence, the client PEP is responsible for initially establishing a session with the PDP, using information that is either configured in the PEP or determined by some other means. Once the session is established, if the Edge Device receives an RSVP message (1), it generates a request for handling to the PDP (2) that describes the context of the request and carries information about the request. The PDP then responds (3) with a decision to accept or reject the request, and if it is accepted, the Edge Device continues by forwarding the RSVP message out into the network (4).

Each session is maintained by a Keep Alive message that verifies that the session is active in case no message has been received recently. Each RSVP or other request is identified by a Handle, which can be used to associate the response, subsequent unsolicited responses, and clearing.

The protocol messages are extensible to other tasks. They consist of an Op Code identifying if the message is a Request, Response, or other type, followed by self-identifying objects, each containing an object class and version identifier. Each object includes a Class Number that defines what the object is, for example, a Timer object, or a Decision object, plus a Class Type that identifies the subtype or version of the Class that is being used.

Other object classes include Bandwidth allocation Data needed for identifying the resources requested by the user, and Policy objects that can be passed down from the PDP to be included in the RSVP message when it is sent out into the network.

X.2 Comparison of COPS and LDAP for policy

Both COPS and LDAP have been associated with Policy-Based Management; however, they would provide very different functions.

COPS is designed for the client to request a decision from a Policy Decision Point and to interact with the PDP to actively participate in the management of policy and policy-related issues. The PEP that makes the request may have no actual knowledge of policies, and relies on the PDP to make decisions based on its knowledge of policies. The protocol allows the PEP to pass information about the request to the PDP, and the PDP to pass back a decision to allow or reject the request.

LDAP is designed for the client to request a directory record from a directory. The function for using the record is dependent on the client, which must be capable of understanding the retrieved record and deciding how to use the information. The server must be capable of finding the correct record based on information in the request, which may involve a search function, or retrieval of multiple records.

Both COPS and LDAP could be used in the context of RSVP admission control. COPS would be used between the PEP and PDP to forward a request for policy-based analysis. LDAP would be used between the PDP and a Directory Server to retrieve policy records associated with the originating and destination addresses for the RSVP request. The PDP would then make a decision based on the retrieved policy information, and use COPS to pass that decision back to the PEP. See Figure X.2.

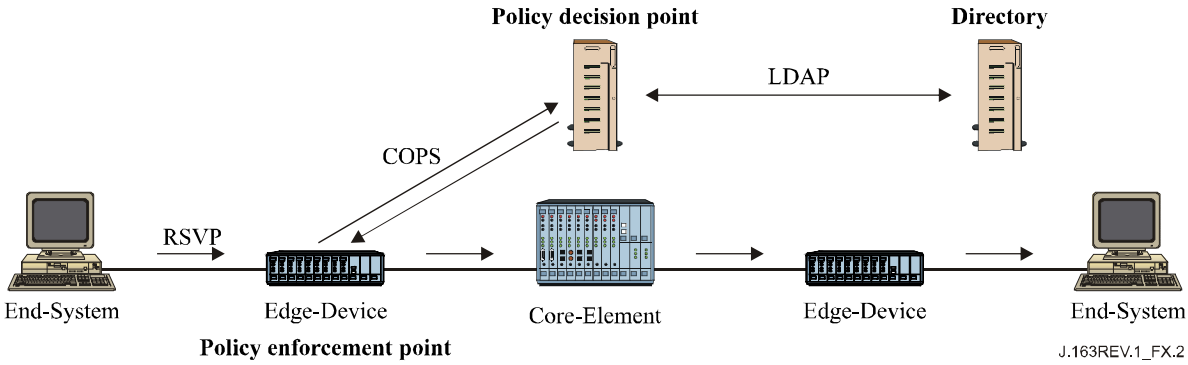


Figure X.2/J.163 – COPS and LDAP model

Appendix XI

RSVP (Resource Reservation Protocol)

XI.1 RSVP procedures and principles

This appendix provides a brief description of RSVP procedures and principles. RSVP is currently defined in IETF RFC 2205. See Figure XI.1.

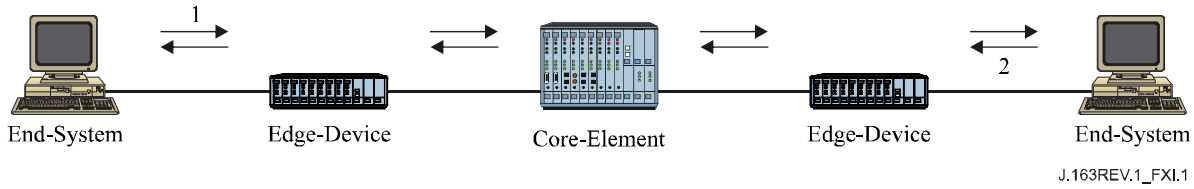


Figure XI.1/J.163 – RSVP

RSVP was developed in the IETF for resource reservation to support information flows across the Internet. Some of the main characteristics of RSVP are:

- resource reservation hop by hop to support end-to-end information flows;
- state information kept at every participating router;
- non-participating routers treat RSVP messages like normal IP packets;
- soft state-reservation must be refreshed periodically or it automatically cancels;
- request driven – an initial PATH message establishes state in the router. A RESV message from the receiver actually results in reservation of resources.

In RSVP, the source initiates a session by sending out a PATH message (1). This is routed through the network based on its destination address (which may be multicast) and creates a flow state at every RSVP-supporting router that it passes through. The PATH message is routed using the same procedures as other IP packets with that destination address, so that it duplicates the route to be followed by data packets. As it progresses, it records the address of the last RSVP-capable router, and this is added to the state information at the next router.

At the receiving end, the receiver joins the session by sending out a RESV message (2) that identifies a flow or flows that this receiver wishes to receive out of the different flows supported in the session. The RESV message traces back the sequence followed by the PATH message, using the records of the last RSVP-capable router, and causes resources to be reserved at each hop. If multiple RESV messages are received at the same router, they may be merged into a singe RESV message with combined resource reservation request.

The process requires state establishment at multiple internal nodes and resource reservation at the same nodes. It establishes a fixed path for the information flow. However, it ensures that resources have been allocated at all RSVP-supporting points in the path.

XI.2 RSVP flowspec

An elementary RSVP reservation request consists of a "flowspec" together with a "Filter-Spec"; this pair is called a "flow descriptor". The flowspec specifies a desired QoS. The filterspec, together with a session specification, defines the set of data packets – the "flow" – to receive the QoS defined by the flowspec. The flowspec is used to set parameters in the node's packet scheduler or other link layer mechanism, while the Filter-Spec is used to set parameters in the packet classifier.

Data packets that are addressed to a particular session but do not match any of the Filter-Specs for that session are handled as best effort traffic.

The flowspec in a reservation request will generally include a service class and two sets of numeric parameters:

1) an "Rspec" (R for "reserve") that defines the desired QoS; and

2) a "Tspec" (T for "traffic") that describes the data flow.

It is important to note that the formats and contents of Tspecs and Rspecs are determined by the integrated service models IETF RFC 2210 defined in the intserv working group of the IETF, and are generally opaque to RSVP itself. RSVP defines the signalling mechanism, and not the traffic model.

Appendix XII

TCP considerations

This Recommendation defines an interface between a Gate Controller (GC) and a Cable modem Termination System (CMTS) to be used for gate authorization, which basically supports a transaction based protocol where each transaction is independent. TCP may be used as a transport for this messaging. However, there were concerns raised about the performance implications of using TCP. This appendix examines a few of these concerns and proposes some potential solutions that can provide an acceptable transport through implementation optimizations and tuning of the TCP implementation.

The design of the network should support the desired degree of reliability and real-time performance.

XII.1 Requirements

Let us first consider requirements on the transport protocol for the interaction between the GC and CMTS:

1) Reliable message delivery for messages exchanged between the GC and CMTS is required.
2) The message exchange should have low latency (of the order of milliseconds), in the normal case (without packet loss). We also need it to have reasonably low latency even under packet loss (of the order of tens of milliseconds).
3) We want multiple requests to be outstanding concurrently. This is because multiple call set-ups are likely to be in progress concurrently.
4) If there is likely to be head-of-the-line (HOL) blocking, this should be avoided.
5) There is likely to be a long-standing association (at least of the order of several minutes) between the GC and the CMTS. However, when there is a failure of a GC, the process of establishing a new connection to the CMTS should not take excessive time. This is especially true when the establishment of a new connection occurs during the time that a call is being set up.

XII.2 Recommended changes

Briefly, the changes we recommend to a vanilla TCP implementation are the following:

1) Modify the time-out mechanism for connection establishment (make it more aggressive).
2) Allow for a larger window after connection establishment.
3) Have multiple TCP connections per GC-CMTS pair to work around potential HOL problems (e.g., use them on a round-robin basis).
4) Lower the 500 ms granularity of the time-out.
5) Disable Nagle's algorithm on the transmit end so as to reduce the latency.
6) Have a non-blocking interface between the application and the TCP stack.

The remainder of this appendix gives details of how these may be implemented.

XII.3 TCP connection establishment impacting post-dial delay

TCP connection establishment uses a three-way handshake as follows (see Figure XII.1).

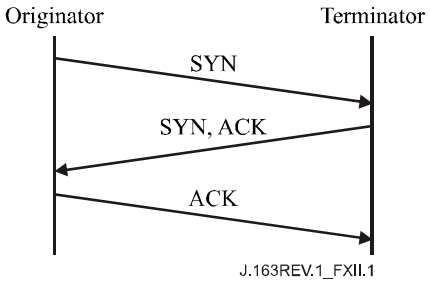


Figure XII.1/J.163 – TCP connection establishment

TCP retransmits segments assumed to be lost based on a round-trip time estimate, A, and a mean deviation, D, from A. The retransmission timeout value (RTO) is generally calculated using the formula:

$$RTO = A + 4D$$

but the initial RTO is calculated using the formula:

$$RTO = A + 2D$$

where A and D are initialized to 0 and 3 seconds respectively. When a retransmission occurs, an exponential backoff using a multiplier of 2 is applied to the current RTO value. Thus, for the first segment, the RTO is calculated as:

$$RTO = 0 + 2 \times 3 = 6$$

Thus, if the initial SYN segment is lost, a retransmission will not occur until 6 seconds later. At that time, RTO will be calculated as:

$$RTO = 0 + 4 \times 3 = 12$$

and an exponential backoff of 2 applied, leading to a new retransmission timeout value of 24 seconds. Thus, should the retransmission be lost as well, a total of 30 seconds will have elapsed before the third retransmission occurs.

The importance of this problem entirely depends on the frequency with which GC $\rightarrow$ CMTS connection establishment falls during the post-dial-delay period. In the currently envisioned scenarios, this occurrence should be very much the exception rather than the rule. The connection set-up delay impacting the post-dial delay is an important reason to avoid having connection establishment in the post-dial-delay period. Diffserv marking of the packets to reduce both latency and loss probability, analogous to what is done with routing traffic today, could be used to reduce connection set-up delays due to lost packets.

XII.4 Need low latency for packets between the GC and CMTS, even under loss

Requirement (2), which deals with recovery of packet loss, needs a few remedies available for TCP to recover from loss quickly. When there are only a few packets being transmitted, and the receiver is unable to generate a sufficient number of duplicate ACKs, the recovery from packet loss is from a retransmission time-out. The TCP retransmission algorithm is based on a smoothed average of the observed round-trip time (RTT), A, and a smoothed average of the mean deviation in RTT. As described above, the retransmission time-out value is then set to:

$$RTO = A + 4D$$

and if the timer fires, the segment in question is retransmitted, and RTO is backed off exponentially using a multiplier of 2[9] until an upper limit of 64 seconds for RTO. Once a segment has been passed to TCP, the segment is either transmitted successfully to the destination or the connection is closed after some period of time has passed (generally a large period of time, e.g., 2 to 9 minutes).

While the above retransmission strategy is deemed desirable, we believe it has two (related) problems for the interface considered:

1) If the segment is not delivered successfully within a small period of time, the call that is in the process of being set up will most likely be abandoned and the transaction should therefore be able to be aborted.

2) The 64-second cap on the retransmission timeout is ill-suited for real time communication and should be set much lower.

A separate, but related issue is that of the granularity of RTO. While the TCP specification itself does not specify the granularity of RTO, it is very common to have a granularity of 500 ms in commercial operating systems. Thus, a lost segment will generally not be detected within less than 500 ms, and two lost segments will not be detected within less than 500 ms + 1000 ms = 1.5 seconds.

To recover rapidly from packet loss in a sequence of packets (without having to depend on multiple duplicate ACKs to trigger fast retransmit or having to wait for the RTO timer to fire), it may be desirable to implement TCP-SACK, which aids recovery even if the fast-retransmit threshold is not reached. It is also recommended that the TCP implementation use a smaller timer granularity (possibly less than 500 milliseconds).

XII.5 Head of line blocking

Head of line blocking refers to the fact, that TCP provides an in-order data delivery service where a lost segment can block later segments from being delivered to the application. Thus, if segments 1 and 2 are sent from A to B, and segment 1 is lost, segment 2 cannot be delivered to the application until segment 1 has been successfully retransmitted.

For the interface considered, this head of line blocking can probably be overcome reasonably well by having multiple TCP connections established between the GC and CMTS, and then use the set of TCP connections in, for example, a round-robin fashion for the transactions. Thus, if a segment is lost on one connection, it will not affect segments, i.e., transactions sent on other connections.

The downside to this approach is that a lost segment is not likely to be retransmitted until its retransmission timer fires (as opposed to a duplicate ACK being received), since there should not be any additional segments to transmit until then.

XII.6 TCP slow start

TCP's ability to start transmitting a stream of data packets is sometimes limited by the TCP slow start mechanism, especially when the stream is a small number (greater than 1) of data packets. It is desirable to choose an initial window that is larger than 1 (both at the beginning of the life of the connection as well as after congestion recovery from a single packet loss). Choosing an initial window size of 2 to 4 MSS is considered desirable. It is important however to ensure that this initial window not exceed 4 MSS, because of the potential to cause congestion itself.

---

[9] TCP furthermore uses duplicate ACKs to trigger retransmission of potentially lost segments; however, we will ignore that for this part of the discussion.

XII.7 Delaying of packets: Nagle's algorithm

TCP/IP was originally designed for supporting multiple user sessions over a slow network. In order to optimize network utilization, the Nagle algorithm was introduced for keyboard input users. Essentially, this algorithm delays the transmission of a packet until a sufficiently large transmit buffer is accumulated or until a certain period of time (usually around 200 milliseconds) elapses.

Due to the real time nature of this traffic, it is advisable to disable the Nagle algorithm for GC-CMTS communication. On most Unix based platforms, Nagle's algorithm can be disabled by issuing the following system call on the socket's file descriptor:

Example 1: Setting the TCP_NODELAY Option

```
/* set TCP No-delay flag (disable Nagle algorithm) */
int flag = 1;
setsockopt(fd, IPPROTO_TCP, TCP_NODELAY, &flag,
               sizeof(flag));
```

Most other languages and platforms have a similar feature to disable the Nagle algorithm, usually known as the TCP_NODELAY option.

XII.8 Non-blocking interface

By default, most operating systems provide a blocking interface for TCP/IP sockets. Although it may allow for an improved error recovery scheme, it has an impact on the performance of the communication channel.

Essentially, a system call such as send() with blocking interface never returns until the operating system confirms that the message was successfully stored in the transmit buffer.

It may be desirable to use a non-blocking interface in order to improve performance and to support asynchronous events using the select() function call on a UNIX based architecture. A non-blocking socket interface can be set up by using the following call on the newly created socket.

Example 2: Setting the O_NONBLOCK Option

```
/* set the socket to non blocking */
fcntl( fd, F_SETFL, O_NONBLOCK );
```

Most other languages and platforms have a similar feature.

Appendix XIII

Incompatible gate parameter change for NCS call for embedded MTA

See Figure XIII.1.

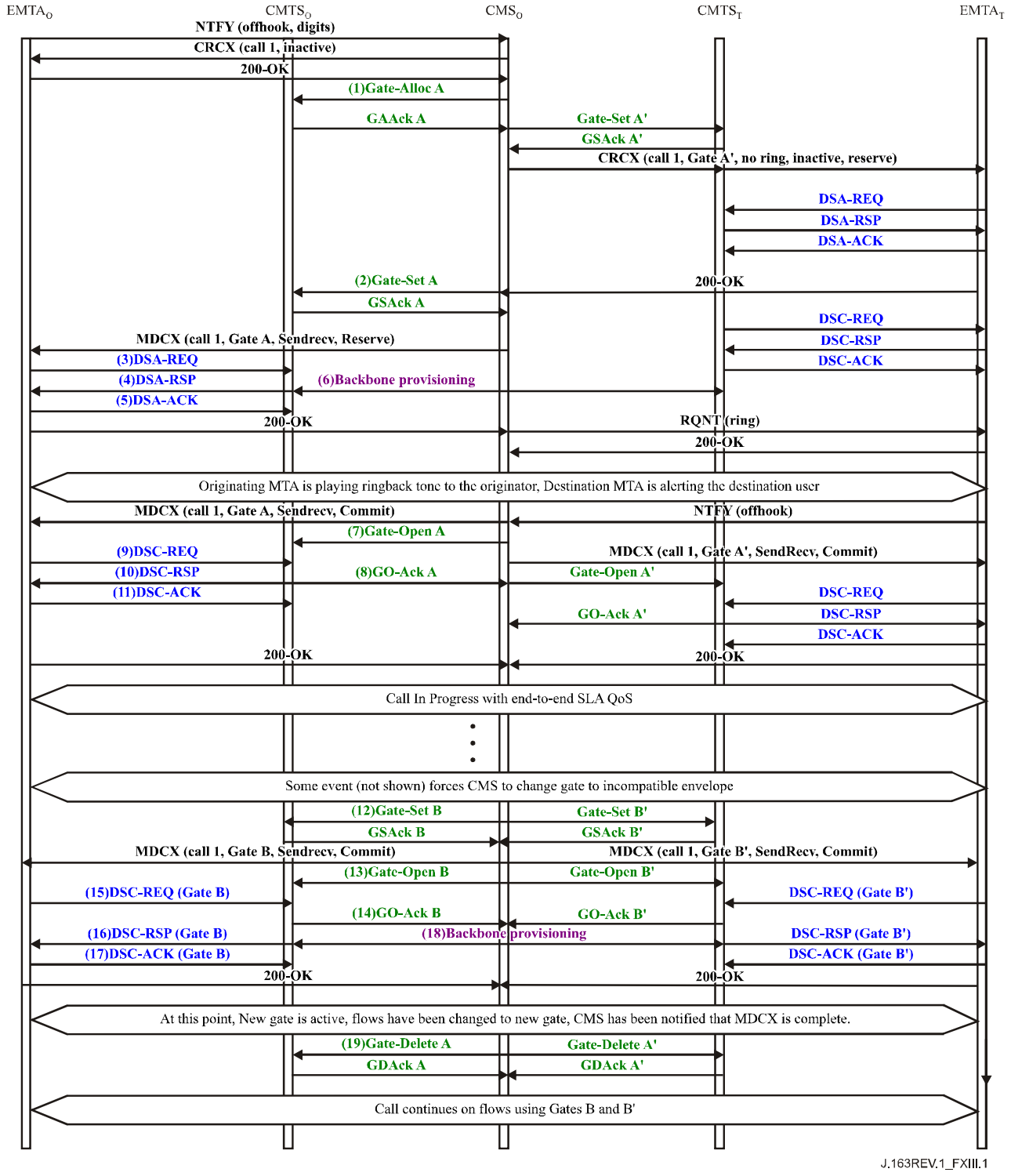


Figure XIII.1/J.163 – On-net to on-net embedded NCS call

1) CMSo, upon receipt of signalling information from EMTAo, checks the current resource consumption of EMTAo by consulting CMTSo.

GATE-ALLOC

TransactionID		3176	
Subscriber		EMTAo	Request new Gate ID and for total resources in use by this client.
Activity-Count		32	Maximum connections allowed by client – expected to be a larger number than ever needed.

CMTSo checks current resource usage by EMTAo, and responds telling the number of connections active.

GATE-ALLOC-ACK

TransactionID		3176	
Subscriber		EMTAo	Request for total resources in use by this client.
Gate-ID		37125	Identifier for allocated Gate A.
Activity Count		1	Total connections established by this client.

2) CMSo, upon further signalling exchanges, gives CMTSo authorization to admit the new connection.

GATE-SET

Transaction ID		3193	Unique Transaction ID for this message exchange.
Subscriber		EMTAo	Request for total resources in use by this client.
Gate-ID		37125	Identifier for allocated Gate A.
Remote-Gate-Info	CMTS Address	CMSo	Information needed to perform gate coordination. Note that CMS has given itself as the entity for exchanging gate coordination messages.
	CMTS Port	2052	
	Remote Gate-ID	8095	
	Security Key	<key>	
	Flag	No-Send-Gate-Open	
Event-Generation-Info	RKS-Addr	RKS	Address of Record Keeping Server.
	RKS-Port	3288	Port on Record Keeping Server.
	Billing Correlation ID	<id>	Opaque data that will be passed to RKS when resources are committed.

GATE-SET

Gate-Spec	Direction	up	
	Protocol	UDP	The protocol, Destination Address, Source Address, and Destination Port quadruple are used for QoS classifiers.
	Source Address	EMTAo	
	Destination Address	EMTAt	
	Source port	7820	
	Destination port	8422	
	DSCP	6	Packet Type value for upstream packets.
	T1	180000	Maximum time between reserve and commit.
	T2	2000	Maximum time for gate coordination to complete.
	r	12000	These are the maximum bandwidth parameters that EMTAo is authorized to request for this conversation.
	b	120	
	p	12000	
	m	120	
	M	120	
	R	12000	
	S	0	
Gate-Spec	Direction	down	
	Protocol	UDP	The protocol, Destination Address, Source Address, and Destination Port quadruple are used for QoS classifiers.
	Source Address	EMTAt	
	Destination Address	EMTAo	
	Source port	8420	
	Destination port	7822	
	DSCP	9	Packet Type value for downstream packets.
	T1	180000	Maximum time between reserve and commit.
	T2	2000	Maximum time for gate coordination to complete.
	r	12000	These are the maximum bandwidth parameters that EMTAo is authorized to request for this conversation.
	b	120	
	p	12000	
	m	120	
	M	120	
	R	12000	
	S	0	

CMTSo responds to the Gate Setup command with an acknowledgement.

GATE-SET-ACK

TransactionID		3193	
Subscriber		EMTAo	Request for total resources in use by this client.
Gate-ID		37125	Identifier for allocated Gate A.
Activity Count		1	Total connections established by this client.

3) EMTAo, upon receiving call signalling information, calculates the QoS parameters for the J.112 link. It sends the following DSA-REQ to the CMTS. This message is used to establish both upstream and downstream parameters. The Upstream Unsolicited Grant Size was calculated as 80 bytes of voice payload plus 12 bytes of RTP header plus 2 bytes of PHS tag plus 2 bytes of header checksum plus 5 bytes of J.112 BPI+ information plus 4 bytes of J.112 MAC header plus 4 bytes of CRC. Header suppression indicates the full 42 bytes of Ethernet/IP/UDP header. Contents of the suppressed header is included in the DSA-REQ.

DSA-REQ

TransactionID		1
UpstreamServiceFlow	ServiceFlowReference	1
	QoSParameterSetType	Admitted (2)
	TimeOutAdmitted	200
	ServiceFlowScheduling	UGS (6)
	NominalGrantInterval	10 ms
	ToleratedGrantJitter	2 ms
	GrantsPerInterval	1
	UnsolicitedGrantSize	109
DownstreamServiceFlow	ServiceFlowReference	2
	QoSParameterSetType	Admitted (2)
	TimeOutAdmitted	200
	TrafficPriority	5
	MaximumSustainedRate	12000
UpstreamPacketClassification	ServiceFlowReference	1
	PacketClassifierReference	1
	ClassifierPriority	150
	ClassifierActivationState	Inactive (0)
	IPSourceAddress	EMTAo
	IPSourcePort	7820
	IPDestinationAddress	EMTAt
	IPDestinationPort	8422
	IPProtocol	UDP (17)

DSA-REQ

DownstreamPacketClassification	ServiceFlowReference	2
	PacketClassifierReference	2
	ClassifierPriority	150
	ClassifierActivationState	Inactive (0)
	IPSourceAddress	EMTAt
	IPSourcePort	8420
	IPDestinationAddress	EMTAo
	IPDestinationPort	7822
	IPProtocol	UDP (17)
PayloadHeaderSuppression	ClassifierReference	1
	ServiceFlowReference	1
	HeaderSuppressionIndex	1
	HeaderSuppressionField	<42bytes>
	HeaderSuppressionMask	<42bits>
	HeaderSuppressionSize	42
	HeaderSuppressionVerify	Verify (0)
AuthorizationBlock		37125
HMAC		

4) The CMTS checks the authorization, by looking for a gate with gate-ID matching the value in AuthBlock, and checks the resources it is required to allocate (e.g., header suppression table space, Service Flow IDs, classifier table space), and installs the classifiers. If the operation is successful, it returns the DSA-RSP message stating the success.

DSA-RSP

TransactionID		1
ConfirmationCode		Success (0)
UpstreamServiceFlow	ServiceFlowReference	1
	ServiceFlowIdentifier	1001
	QoSParameterSetType	Admitted (2)
	TimeOutAdmitted	200
	ServiceFlowScheduling	UGS (6)
	NominalGrantInterval	10 ms
	ToleratedGrantJitter	2 ms
	GrantsPerInterval	1
	UnsolicitedGrantSize	109
DownstreamServiceFlow	ServiceFlowReference	2
	ServiceFlowIdentifier	2001
	QoSParameterSetType	Admitted (2)
	TimeOutAdmitted	200
	TrafficPriority	5
	MaximumSustainedRate	12000

DSA-RSP

UpstreamPacketClassification	ServiceFlowReference	1
	PacketClassifierReference	1
	PacketClassifierIdentifier	3001
	ClassifierPriority	150
	ClassifierActivationState	Inactive (0)
	IPSourceAddress	EMTAo
	IPSourcePort	7820
	IPDestinationAddress	EMTAt
	IPDestinationPort	8422
	IPProtocol	UDP (17)
DownstreamPacketClassification	ServiceFlowReference	2
	PacketClassifierReference	2
	PacketClassifierIdentifier	3002
	ClassifierPriority	150
	ClassifierActivationState	Inactive (0)
	IPSourceAddress	EMTAt
	IPDestinationAddress	EMTAo
	IPSourcePort	8420
	IPDestinationPort	7822
	IPProtocol	UDP (17)
HMAC		

5) Upon receipt of the DSA-RSP, the CM acknowledges receipt with a DSA-ACK message.

DSA-ACK

TransactionID		1
ConfirmationCode		Success (0)
HMAC		

6) Simultaneous with message No. 4, the CMTS initiates any required backbone reservations for the requested quality of service. The content of this message is dependent on the particular backbone algorithms in use, and is outside the scope of this Recommendation. The backbone router sends to the CMTS any required notification that the reservation is successful.

7) The CMS sends the gate open message to the CMTS to inform it that the resources should be committed. If the CMTS does not receive the DSC-REQ from EMTAo within a short time, it should revoke the gate authorization.

GATE-OPEN

Transaction ID		72	Identifier to match this message with its response.
Gate ID		37125	Gate-ID A at CMTS.
HMAC			Security checksum for this message.

8) The CMTS responds to the GATE-OPEN with:

GATE-OPEN-ACK

Transaction ID		72	Identifier to match this message with its response.
HMAC			Security checksum for this message.

9) In response to signalling messages that indicate the call has been answered, EMTAo uses the Annex E to Annex B/J.112 interface to activate the admitted resources. This is done via a DSC-REQ to the CMTS.

DSC-REQ

TransactionID		2
UpstreamServiceFlow	ServiceFlowIdentifier	1001
	QoSParameterSetType	Admitted + Activated (6)
	TimeOutActive	10
	ServiceFlowScheduling	UGS (6)
	NominalGrantInterval	10 ms
	ToleratedGrantJitter	2 ms
	GrantsPerInterval	1
	UnsolicitedGrantSize	109
DownstreamServiceFlow	ServiceFlowIdentifier	2001
	QoSParameterSetType	Admitted + Activated (6)
	TimeOutActive	10
	TrafficPriority	5
	MaximumSustainedRate	12000
UpstreamPacketClassification	ServiceFlowIdentifier	1001
	PacketClassifierIdentifier	3001
	ClassifierChangeAction	Replace (1)
	ClassifierPriority	150
	ClassifierActivationState	Active (1)
	IPSourceAddress	EMTAo
	IPSourcePort	7820
	IPDestinationAddress	EMTAt
	IPDestinationPort	8422
	IPProtocol	UDP (17)

DSC-REQ

DownstreamPacketClassification	ServiceFlowIdentifier	2001
	PacketClassifierIdentifier	3002
	ClassifierChangeAction	Replace (1)
	ClassifierPriority	150
	ClassifierActivationState	Active (1)
	IPSourceAddress	EMTAt
	IPDestinationAddress	EMTAo
	IPSourcePort	8420
	IPDestinationPort	7822
	IPProtocol	UDP (17)
AuthorizationBlock		37125
HMAC		

10) The CMTS sends a DSC-RSP message showing the operation was successful.

DSC-RSP

TransactionID		2
ConfirmationCode		Success (0)
HMAC		

11) The CM sends DSC-ACK message to indicate that the DSC-RSP has been received and agreed.

DSC-ACK

TransactionID		2
ConfirmationCode		Success (0)
HMAC		

12) For some reason not shown (in this example, a port number change and codec change to 729E with 30 ms packets), CMSo wishes to modify the resource parameters of the call to be incompatible with the resource parameters of Gate A (37125). CMSo, upon further signalling exchanges, gives CMTSo authorization to admit the new connection.

GATE-SET

Transaction ID		95	Unique Transaction ID for this message exchange.
Subscriber		EMTAo	Request for total resources in use by this client.
Activity Count		32	
Remote-Gate-Info	CMTS Address	CMSo	Information needed to perform gate coordination. Note that CMS has given itself as the entity for exchanging gate coordination messages.
	CMTS Port	2052	
	Remote Gate-ID	8095	
	Security Key	<key>	
	Flag	No-Send-Gate-Open,	

GATE-SET

Event-Generation-Info	RKS-Addr	RKS	Address of Record Keeping Server.
	RKS-Port	3288	Port on Record Keeping Server.
	Billing Correlation ID	<id>	Opaque data that will be passed to RKS when resources are committed.
Gate-Spec	Direction	up	
	Protocol	UDP	The protocol, Destination Address, Source Address, and Destination Port quadruple are used for QoS classifiers.
	Source Address	EMTAo	
	Destination Address	EMTAt	
	Source port	7820	
	Destination port	8632	
	DSCP	6	Packet Type value for upstream packets.
	T1	180000	Maximum time between reserve and commit.
	T2	2000	Maximum time for gate coordination to complete.
	r	2833	These are the maximum bandwidth parameters that EMTAo is authorized to request for this conversation.
	b	85	
	p	2833	
	m	85	
	M	85	
	R	2833	
	S	0	
Gate-Spec	Direction	down	
	Protocol	UDP	The protocol, Destination Address, Source Address, and Destination Port quadruple are used for QoS classifiers.
	Source Address	EMTAt	
	Destination Address	EMTAo	
	Source port	8630	
	Destination port	7822	
	DSCP	9	Packet Type value for downstream packets.
	T1	180000	Maximum time between reserve and commit.
	T2	2000	Maximum time for gate coordination to complete.
	r	2833	These are the maximum bandwidth parameters that EMTAo is authorized to request for this conversation.
	b	85	
	p	2833	
	m	85	
	M	85	
	R	2833	
	S	0	

CMTSo responds to the Gate Setup command with an acknowledgement.

GATE-SET-ACK

TransactionID		95	
Subscriber		EMTAo	Request for total resources in use by this client.
Gate-ID		38205	Identifier for newly allocated Gate B.
Activity Count		32	

13) The CMS sends the gate open message to the CMTS to inform it that the resources should be committed. If the CMTS does not receive the DSC-REQ from EMTAo within a short time, it should revoke the gate authorization.

GATE-OPEN

Transaction ID		143	Identifier to match this message with its response.
Gate ID		38205	Gate-ID B at CMTS.
HMAC			Security checksum for this message.

14) The CMTS responds to the GATE-OPEN with:

GATE-OPEN-ACK

Transaction ID		143	Identifier to match this message with its response.
HMAC			Security checksum for this message.

15) EMTAo, upon receiving call signalling information, calculates the QoS parameters for the J.112 link. It sends the following DSC-REQ to the CMTS. This message is used to establish both upstream and downstream parameters. The Upstream Unsolicited Grant Size was calculated as 30 bytes of voice payload plus 12 bytes of RTP header plus 2 bytes of PHS tag plus 2 bytes of header checksum plus 5 bytes of J.112 BPI+ information plus 4 bytes of J.112 MAC header plus 4 bytes of CRC. Header suppression indicates the 42 bytes of Ethernet/IP/UDP header. Contents of the suppressed header is included in the DSC-REQ.

DSC-REQ

TransactionID		2004
UpstreamServiceFlow	ServiceFlowIdentifier	1001
	QoSParameterSetType	Admitted + Activated (6)
	TimeOutAdmitted	200
	ServiceFlowScheduling	UGS (6)
	NominalGrantInterval	30 ms
	ToleratedGrantJitter	2 ms
	GrantsPerInterval	1
	UnsolicitedGrantSize	59

DSC-REQ

DownstreamServiceFlow	ServiceFlowIdentifier	2001
	QoSParameterSetType	Admitted + Activated (6)
	TimeOutActive	10
	TrafficPriority	5
	MaximumSustainedRate	2833
Upstream Classifier	ServiceFlowIdentifier	1001
	PacketClassifierIdentifier	3001
	ClassifierChangeAction	Replace (1)
	ClassifierPriority	150
	ClassifierActivationState	Active (1)
	IPSourceAddress	EMTAo
	IPSourcePort	7820
	IPDestinationAddress	EMTAt
	IPDestinationPort	8632
	IPProtocol	UDP (17)
Downstream Classifier	ServiceFlowIdentifier	2001
	PacketClassifierIdentifier	3002
	ClassifierChangeAction	Replace (1)
	ClassifierPriority	150
	ClassifierActivationState	Active (1)
	IPSourceAddress	EMTAt
	IPDestinationAddress	EMTAo
	IPSourcePort	8630
	IPDestinationPort	7822
	IPProtocol	UDP (17)
AuthorizationBlock		38205
HMAC		

16) Upon receipt of the DSC-REQ from the EMTA, the CMTS sends a DSC-RSP to the EMTA.

DSC-RSP

TransactionID		2004
ConfirmationCode		Success (0)
HMAC		

17) Upon receipt of the DSC-RSP from the CMTS, the EMTA sends a DSC-ACK to the CMTS.

DSC-ACK

TransactionID		2004
ConfirmationCode		Success (0)
HMAC		

18) Simultaneous with message No. 16, the CMTS initiates any required backbone reservations for the requested quality of service. The content of this message is dependent on the particular backbone algorithms in use, and is outside the scope of this Recommendation. The backbone router sends to the CMTS any required notification that the reservation is successful.

19) Once the CMSo received the 200 OK from the MTA – signalling that the call is now successfully transferred to the new Gate B, the CMSo issues a Gate Delete message for the now unused gate A.

GATE-DELETE

TransactionID	143	Identifier to match this message with its response.
Gate ID	37125	Gate-ID A at CMTS.

CMTSo responds to the Gate Setup command with an acknowledgement.

GATE-DELETE-ACK

TransactionID		95	
Gate-ID		37125	Identifier Gate A.

Appendix XIV

Incompatible gate parameter change for NCS call for embedded MTA

The following example (Figure XIV.1) illustrates how Call Waiting is handled when using NCS signalling and CM initiated DQoS RSVP messaging. The call flow below assumes that a call is already in progress between MTAo and MTA$_{T1}$ using GateId #1 (37125) and ServiceFlows #1. The 2nd connection for MTA$_{T2}$ opens a new Gate (38205) and ServiceFlows and uses the ResourceId (472) returned from the initial call to indicate to the CMTS to share the underlying bandwidth for these 2 service flows.

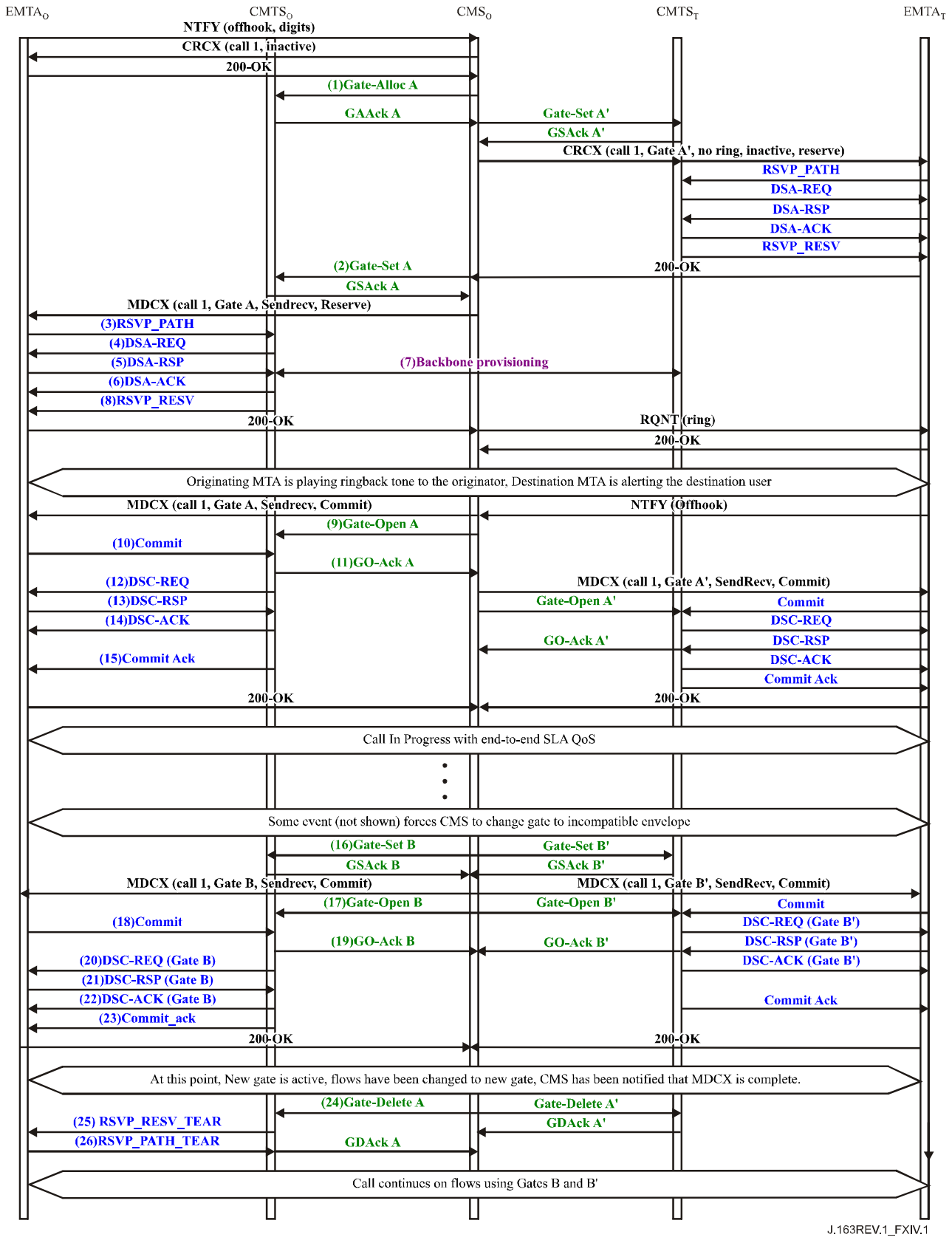


Figure XIV.1/J.163 – On-net to on-net embedded NCS call

1) CMSo, upon receipt of signalling information from EMTAo, checks the current resource consumption of EMTAo by consulting CMTSo.

GATE-ALLOC

TransactionID		3176	
Subscriber		EMTAo	Request new Gate ID and for total resources in use by this client.
Activity-Count		32	Maximum connections allowed by client – expected to be a larger number than ever needed.

CMTSo checks current resource usage by EMTAo, and responds telling the number of connections active.

GATE-ALLOC-ACK

TransactionID		3176	
Subscriber		EMTAo	Request for total resources in use by this client.
Gate-ID		37125	Identifier for allocated Gate A.
Activity Count		1	Total connections established by this client.

2) CMSo, upon further signalling exchanges, gives CMTSo authorization to admit the new connection.

GATE-SET

Transaction ID		3193	Unique Transaction ID for this message exchange.
Subscriber		EMTAo	Request for total resources in use by this client.
Gate-ID		37125	Identifier for allocated Gate A.
Remote-Gate-Info	CMTS Address	CMSo	Information needed to perform gate coordination. Note that CMS has given itself as the entity for exchanging gate coordination messages.
	CMTS Port	2052	
	Remote Gate-ID	8095	
	Security Key	<key>	
	Flag	No-Send-Gate-Open	
Event-Generation-Info	RKS-Addr	RKS	Address of Record Keeping Server.
	RKS-Port	3288	Port on Record Keeping Server.
	Billing Correlation ID	<id>	Opaque data that will be passed to RKS when resources are committed.

GATE-SET

Gate-Spec	Direction	up	
	Protocol	UDP	The protocol, Destination Address, Source Address, and Destination Port quadruple are used for QoS classifiers.
	Source Address	EMTAo	
	Destination Address	EMTAt	
	Source port	7820	
	Destination port	8422	
	DSCP	6	Packet Type value for upstream packets.
	T1	180000	Maximum time between reserve and commit.
	T2	2000	Maximum time for gate coordination to complete.
	rb	12000	These are the maximum bandwidth parameters that EMTAo is authorized to request for this conversation.
	br	12000	
	p	12000	
	m	120	
	M	120	
	R	12000	
	S	0	
Gate-Spec	Direction	down	
	Protocol	UDP	The protocol, Destination Address, Source Address, and Destination Port quadruple are used for QoS classifiers.
	Source Address	EMTAt	
	Destination Address	EMTAo	
	Source port	8420	
	Destination port	7822	
	DSCP	9	Packet Type value for downstream packets.
	T1	180000	Maximum time between reserve and commit.
	T2	2000	Maximum time for gate coordination to complete.
	rb	12000	These are the maximum bandwidth parameters that EMTAo is authorized to request for this conversation.
	br	12000	
	p	12000	
	m	120	
	M	120	
	R	12000	
	S	0	

CMTSo responds to the Gate Setup command with an acknowledgement.

GATE-SET-ACK

TransactionID		3193	
Subscriber		EMTAo	Request for total resources in use by this client.
Gate-ID		37125	Identifier for allocated Gate A.
Activity Count		1	Total connections established by this client.

3) EMTAo, upon receiving call-signalling information, calculates the QoS parameters for the DOCSIS 1.1 link, and sends an RSVP message to the terminating MTA.

RSVP-PATH

Session-Object	Protocol	UDP	The parameters identify the RSVP session, match the authorization previously sent by the GateController, and are also used for QoS classifiers.
	Destination Address	EMTAt	
	Destination port	7820	
Sender Templ	Source Address	EMTAo	
	Source port	8422	
Sender-Tspec	b	120	These give the Least-Upper-Bound for all of the individual traffic parameters for the two separate possible flows. This is a standard RSVP object, which will be interpreted by all intermediate routers in the path between the MTA and CMTS.
	r	12000	
	p	12000	
	m	120	
	M	120	
	Hdr Suppression	40	
	VAD	off	
Gate-ID		37125	Identity of gate that authorizes this request.
Forward Rspec	R	12000	Rspec that corresponds to the immediately preceding Sender Tspec.
	S	0	
Reverse-Session	Protocol	UDP	New RSVP objects that provide the CMTS with sufficient information to calculate downstream traffic parameters and to generate an RSVP-PATH message for the downstream flow.
	Destination Addr	EMTAo	
	Destination port	7822	
Reverse-Sender Templ	Source Address	EMTAt	
	Source port	8420	
Reverse-Sender-Tspec	b	120	Negotiated traffic parameters for the new CODEC being requested for this call. The CMTS calculates the actual downstream QoS parameters using these Tspec and Rspec parameters. This is a new RSVP object, which will be ignored by intermediate routers.
	r	12000	
	p	12000	
	m	120	
	M	120	
	Hdr Suppression	0	
	VAD	off	
Reverse-Rspec	R	12000	
	S	0	

4) CMTS, upon receiving RSVP message, checks the authorization, by looking for a gate with gate-ID matching the value in Gate-ID, and checks the resources it is required to allocate (e.g., header suppression table space, Service Flow IDs, classifier table space), then it calculates the QoS parameters for the DOCSIS 1.1 link. It uses the DOCSIS RFI Annex E to Annex B/J.112 interface to the Cable Modem to send the following DSA-REQ to the EMTAo. This message is used to establish both upstream and downstream parameters. The Upstream Unsolicited Grant Size was calculated as 80 bytes of voice payload plus 12 bytes of RTP header plus 2 bytes of PHS tag plus 2 bytes of header checksum plus 5 bytes of DOCSIS BPI+ information plus 4 bytes of DOCSIS MAC header plus 4 bytes of CRC. Header suppression indicates the full 42 bytes of Ethernet/IP/UDP header. Contents of the suppressed header is included in the DSA-REQ.

DSA-REQ

TransactionID		1
UpstreamServiceFlow	ServiceFlowReference	1
	QoSParameterSetType	Admitted (2)
	TimeOutAdmitted	200
	ServiceFlowScheduling	UGS (6)
	NominalGrantInterval	10 ms
	ToleratedGrantJitter	2 ms
	GrantsPerInterval	1
	UnsolicitedGrantSize	109
DownstreamServiceFlow	ServiceFlowReference	2
	QoSParameterSetType	Admitted (2)
	TimeOutAdmitted	200
	TrafficPriority	5
	MaximumSustainedRate	12000
UpstreamPacketClassification	ServiceFlowReference	1
	PacketClassifierReference	1
	ClassifierPriority	150
	ClassifierActivationState	Inactive (0)
	IPSourceAddress	EMTAo
	IPSourcePort	7820
	IPDestinationAddress	EMTAt
	IPDestinationPort	8422
	IPProtocol	UDP (17)
DownstreamPacketClassification	ServiceFlowReference	2
	PacketClassifierReference	2
	ClassifierPriority	150
	ClassifierActivationState	Inactive (0)
	IPSourceAddress	EMTAt
	IPSourcePort	8420
	IPDestinationAddress	EMTAo
	IPDestinationPort	7822
	IPProtocol	UDP (17)

DSA-REQ

PayloadHeaderSuppression	ClassifierReference	1
	ServiceFlowReference	1
	HeaderSuppressionIndex	1
	HeaderSuppressionField	<42bytes>
	HeaderSuppressionMask	<42bits>
	HeaderSuppressionSize	42
	HeaderSuppressionVerify	Verify (0)
AuthorizationBlock		37125
HMAC		

5) The EMTAo checks the admission, and installs the classifiers. If the operation is successful, it returns the DSA-RSP message stating the success.

DSA-RSP

TransactionID		1
ConfirmationCode		Success (0)
UpstreamServiceFlow	ServiceFlowReference	1
	ServiceFlowIdentifier	1001
	QoSParameterSetType	Admitted (2)
	TimeOutAdmitted	200
	ServiceFlowScheduling	UGS (6)
	NominalGrantInterval	10 ms
	ToleratedGrantJitter	2 ms
	GrantsPerInterval	1
	UnsolicitedGrantSize	109
DownstreamServiceFlow	ServiceFlowReference	2
	ServiceFlowIdentifier	2001
	QoSParameterSetType	Admitted (2)
	TimeOutAdmitted	200
	TrafficPriority	5
	MaximumSustainedRate	12000
UpstreamPacketClassification	ServiceFlowReference	1
	PacketClassifierReference	1
	PacketClassifierIdentifier	3001
	ClassifierPriority	150
	ClassifierActivationState	Inactive (0)
	IPSourceAddress	EMTAo
	IPSourcePort	7820
	IPDestinationAddress	EMTAt
	IPDestinationPort	8422
	IPProtocol	UDP (17)

DSA-RSP

DownstreamPacketClassification	ServiceFlowReference	2
	PacketClassifierReference	2
	PacketClassifierIdentifier	3002
	ClassifierPriority	150
	ClassifierActivationState	Inactive (0)
	IPSourceAddress	EMTAt
	IPDestinationAddress	EMTAo
	IPSourcePort	8420
	IPDestinationPort	7822
	IPProtocol	UDP (17)
HMAC		

6) Upon receipt of the DSA-RSP, the CMTS acknowledges receipt with a DSA-ACK message.

DSA-ACK

TransactionID		1
ConfirmationCode		Success (0)
HMAC		

7) Simultaneous with message No. 5, the CMTS initiates any required backbone reservations for the requested quality of service. The content of this message is dependent on the particular backbone algorithms in use, and is outside the scope of this Recommendation. The backbone router sends to the CMTS any required notification that the reservation is successful.

8) CMTS acknowledges the success of the reservation by sending RSVP_RESV message back to the EMTAo.

RSVP-RESV

Session-Object	Protocol	UDP	These fields identify the IP flow for which the reservation is being established.
	Destination Address	EMTAt	
	Destination port	7820	
Filter-Spec	Source Address	EMTAo	
	Source port	8422	
Flowspec	br	12000	These fields identify the resources being reserved for this flow.
	rb	12000	
	p	12000	
	m	120	
	M	120	
	R	12000	
	S	0	
ResourceID		472	Resource ID for this reservation.

9) The CMS sends the gate open message to the CMTS to inform it that the resources should be committed. If the CMTS does not receive the DSC-REQ Commit Message from EMTAo within a short time, it should revoke the gate authorization.

GATE-OPEN

Transaction ID		72	Identifier to match this message with its response.
Gate ID		37125	Gate-ID A at CMTS.
HMAC			Security checksum for this message.

10) In response to signalling messages that indicate the call has been answered, EMTAo uses the Annex E to Annex B/J.112 interface Commit Message to activate the admitted resources. This is done via a DSC-REQ DOCSIS 1.1 command to the CMTS.

COMMIT

Session-Object	Protocol	UDP	The protocol, Destination Address, Source Address, and Destination Port must match those of the Gate ID.
	Destination Address	EMTAt	
	Destination port	7820	
Sender Templ	Source Address	EMTAo	
	Source port	8422	
Gate-ID		37125	

11) The CMTS responds to the GATE-OPEN with:

GATE-OPEN-ACK

Transaction ID		72	Identifier to match this message with its response.
HMAC			Security checksum for this message.

12) In response to signalling messages that indicate the call has been answered, CMTSo uses the Annex E to Annex B/J.112 interface to activate the admitted resources. This is done via a DSC-REQ DOCSIS 1.1 command to the EMTAo.

DSC-REQ

TransactionID		2
UpstreamServiceFlow	ServiceFlowIdentifier	1001
	QoSParameterSetType	Admitted + Activated (6)
	TimeOutActive	10
	ServiceFlowScheduling	UGS (6)
	NominalGrantInterval	10 ms
	ToleratedGrantJitter	2 ms
	GrantsPerInterval	1
	UnsolicitedGrantSize	109

DSC-REQ

DownstreamServiceFlow	ServiceFlowIdentifier	2001
	QoSParameterSetType	Admitted + Activated (6)
	TimeOutActive	10
	TrafficPriority	5
	MaximumSustainedRate	12000
UpstreamPacketClassification	ServiceFlowIdentifier	1001
	PacketClassifierIdentifier	3001
	ClassifierChangeAction	Replace (1)
	ClassifierPriority	150
	ClassifierActivationState	Active (1)
	IPSourceAddress	EMTAo
	IPSourcePort	7820
	IPDestinationAddress	EMTAt
	IPDestinationPort	8422
	IPProtocol	UDP (17)
DownstreamPacketClassification	ServiceFlowIdentifier	2001
	PacketClassifierIdentifier	3002
	ClassifierChangeAction	Replace (1)
	ClassifierPriority	150
	ClassifierActivationState	Active (1)
	IPSourceAddress	EMTAt
	IPDestinationAddress	EMTAo
	IPSourcePort	8420
	IPDestinationPort	7822
	IPProtocol	UDP (17)
AuthorizationBlock		37125
HMAC		

13) The EMTAo sends a DSC-RSP message showing the operation was successful.

DSC-RSP

TransactionID		2
ConfirmationCode		Success (0)
HMAC		

14) The CMTS sends DSC-ACK message to indicate that the DSC-RSP has been received and agreed.

DSC-ACK

TransactionID		2
ConfirmationCode		Success (0)
HMAC		

15) The CMTS acknowledges the COMMIT with:

COMMIT-ACK

Session-Object	Protocol	UDP	The protocol, Destination Address, Source Address, and Destination Port quadruple matches Gate ID.
	Destination Address	MTAt2	
	Destination port	7820	
Sender Templ	Source Address	MTAo	
	Source port	8422	
Gate-ID		37125	

16) For some reason not shown (in this example, a port number change and codec change to 729E with 30 ms packets), CMSo wishes to modify the resource parameters of the call to be incompatible with the resource parameters of Gate A (37125). CMSo, upon further signalling exchanges, gives CMTSo authorization to admit the new connection.

GATE-SET

Transaction ID		95	Unique Transaction ID for this message exchange.
Subscriber		EMTAo	Request for total resources in use by this client.
Activity Count		32	
Remote-Gate-Info	CMTS Address	CMSo	Information needed to perform gate coordination. Note that CMS has given itself as the entity for exchanging gate coordination messages.
	CMTS Port	2052	
	Remote Gate-ID	8095	
	Security Key	<key>	
	Flag	No-Send-Gate-Open,	
Event-Generation-Info	RKS-Addr	RKS	Address of Record Keeping Server.
	RKS-Port	3288	Port on Record Keeping Server.
	Billing Correlation ID	<id>	Opaque data that will be passed to RKS when resources are committed.

GATE-SET

Gate-Spec	Direction	up	
	Protocol	UDP	The protocol, Destination Address, Source Address, and Destination Port quadruple are used for QoS classifiers.
	Source Address	EMTAo	
	Destination Address	EMTAt	
	Source port	7820	
	Destination port	8632	
	DSCP	6	Packet Type value for upstream packets.
	T1	180000	Maximum time between reserve and commit.
	T2	2000	Maximum time for gate coordination to complete.
	rb	852833	These are the maximum bandwidth parameters that EMTAo is authorized to request for this conversation.
	br	283385	
	p	2833	
	m	85	
	M	85	
	R	2833	
	S	0	
Gate-Spec	Direction	down	
	Protocol	UDP	The protocol, Destination Address, Source Address, and Destination Port quadruple are used for QoS classifiers.
	Source Address	EMTAt	
	Destination Address	EMTAo	
	Source port	8630	
	Destination port	7822	
	DSCP	9	Packet Type value for downstream packets.
	T1	180000	Maximum time between reserve and commit.
	T2	2000	Maximum time for gate coordination to complete.
	rb	283385	These are the maximum bandwidth parameters that EMTAo is authorized to request for this conversation.
	br	283385	
	p	2833	
	m	85	
	M	85	
	R	2833	
	S	0	

CMTSo responds to the Gate Setup command with an acknowledgement.

GATE-SET-ACK

TransactionID		95	
Subscriber		EMTAo	Request for total resources in use by this client.
Gate-ID		38205	Identifier for newly allocated Gate B.
Activity Count		32	

17) The CMS sends the gate open message to the CMTS to inform it that the resources should be committed. If the CMTS does not receive the DSC-REQ from EMTAo within a short time, it should revoke the gate authorization.

GATE-OPEN

Transaction ID		143	Identifier to match this message with its response.
Gate ID		38205	Gate-ID B at CMTS.
HMAC			Security checksum for this message.

18) EMTAo, upon receiving call-signalling information, calculates the QoS parameters for the DOCSIS 1.1 link, and reserves bandwidth using RSVP messaging. Then when the modify connection with commit is received from the CMSo, it uses the Commit Message to commit the resources.

COMMIT

Session-Object	Protocol	UDP	The protocol, Destination Address, Source Address, and Destination Port must match those of the Gate ID.
	Destination Address	EMTAt	
	Destination port	7820	
Sender Templ	Source Address	EMTAo	
	Source port	86432	
Gate-ID		37126	

RSVP-PATH

Session-Object	Protocol	UDP	The parameters identify the RSVP session, match the authorization previously sent by the GateController, and are also used for QoS classifiers.
	Destination Address	EMTAt	
	Destination port	7820	
Sender Templ	Source Address	EMTAo	
	Source port	8632	
Sender-Tspec	r	2833	These give the Least-Upper-Bound for all of the individual traffic parameters for the two separate possible flows. This is a standard RSVP object, which will be interpreted by all intermediate routers in the path between the MTA and CMTS.
	b	85	
	p	2833	
	m	85	
	M	85	
	Hdr Suppression	40	
	VAD	off	
Gate-ID		37126	Identity of gate that authorizes this request.

RSVP-PATH

Resource ID		472	The resource ID that identifies the previous RSVP link.
Forward Rspec	R	12000	Rspec that corresponds to the immediately preceding Sender Tspec.
	S	0	
Reverse-Session	Protocol	UDP	New RSVP objects that provide the CMTS with sufficient information to calculate downstream traffic parameters and to generate an RSVP-PATH message for the downstream flow.
	Destination Addr	EMTAo	
	Destination port	7822	
Reverse-Sender Templ	Source Address	EMTAt	
	Source port	8420	
Reverse-Sender-Tspec	r	2833	Negotiated traffic parameters for the new CODEC being requested for this call. The CMTS calculates the actual downstream QoS parameters using these Tspec and Rspec parameters. This is a new RSVP object, which will be ignored by intermediate routers.
	b	85	
	p	2833	
	m	85	
	M	85	
	Hdr Suppression	0	
	VAD	off	
Reverse-Rspec	R	2833	
	S	0	

19) The CMTS responds to the GATE-OPEN with:

GATE-OPEN-ACK

Transaction ID		143	Identifier to match this message with its response.
HMAC			Security checksum for this message.

20) CMTSo, upon receiving call-signalling information, calculates the QoS parameters for the DOCSIS 1.1 link. It uses the DOCSIS RFI Annex E to Annex B/J.112 interface to the Cable Modem to send the following DSC-REQ to the EMTAo. This message is used to establish both upstream and downstream parameters. The Upstream Unsolicited Grant Size was calculated as 30 bytes of voice payload plus 12 bytes of RTP header plus 2 bytes of PHS tag plus 2 bytes of header checksum plus 5 bytes of DOCSIS BPI information plus 4 bytes of DOCSIS MAC header plus 4 bytes of CRC. Header suppression indicates the 42 bytes of Ethernet/IP/UDP header. Contents of the suppressed header is included in the DSC-REQ.

DSC-REQ

TransactionID		2004
UpstreamServiceFlow	ServiceFlowIdentifier	1001
	QoSParameterSetType	Admitted + Activated (6)
	TimeOutAdmitted	200
	ServiceFlowScheduling	UGS (6)
	NominalGrantInterval	30 ms
	ToleratedGrantJitter	2 ms
	GrantsPerInterval	1
	UnsolicitedGrantSize	59
DownstreamServiceFlow	ServiceFlowIdentifier	2001
	QoSParameterSetType	Admitted + Activated (6)
	TimeOutActive	10
	TrafficPriority	5
	MaximumSustainedRate	2833
Upstream Classifier	ServiceFlowIdentifier	1001
	PacketClassifierIdentifier	3001
	ClassifierChangeAction	Replace (1)
	ClassifierPriority	150
	ClassifierActivationState	Active (1)
	IPSourceAddress	EMTAo
	IPSourcePort	7820
	IPDestinationAddress	EMTAt
	IPDestinationPort	8632
	IPProtocol	UDP (17)
Downstream Classifier	ServiceFlowIdentifier	2001
	PacketClassifierIdentifier	3002
	ClassifierChangeAction	Replace (1)
	ClassifierPriority	150
	ClassifierActivationState	Active (1)
	IPSourceAddress	EMTAt
	IPDestinationAddress	EMTAo
	IPSourcePort	8630
	IPDestinationPort	7822
	IPProtocol	UDP (17)
AuthorizationBlock		38205
HMAC		

21) Upon receipt of the DSC-REQ from the CMTS, EMTAo sends a DSC-RSP to the CMTS.

DSC-RSP

TransactionID		2004
ConfirmationCode		Success (0)
HMAC		

22) Upon receipt of the DSC-RSP from the EMTAo, the CMTSo sends a DSC-ACK to the EMTAo.

DSC-ACK

TransactionID		2004
ConfirmationCode		Success (0)
HMAC		

23) Simultaneous with message No. 21, the CMTS initiates any required backbone reservations for the requested quality of service. The content of this message is dependent on the particular backbone algorithms in use, and is outside the scope of this Recommendation. The backbone router sends to the CMTS any required notification that the reservation is successful.

24) The CMTS acknowledges the commit with.

COMMIT-ACK

Session-Object	Protocol	UDP	The protocol, Destination Address, Source Address, and Destination Port quadruple matches Gate ID.
	Destination Address	MTAt2	
	Destination port	7820	
Sender Templ	Source Address	MTAo	
	Source port	84632	
Gate-ID		37126	

25) Once the CMSo received the 200 OK from the MTA – signalling that the call is now successfully transferred to the new Gate B, the CMSo issues a Gate Delete message for the now unused gate A.

GATE-DELETE

TransactionID	143	Identifier to match this message with its response.
Gate ID	37125	Gate-ID A at CMTS.

CMTSo responds to the Gate Setup command with an acknowledgement.

GATE-DELETE-ACK

TransactionID		95	
Gate-ID		37125	Identifier Gate A.

26) Upon reception of Gate_Delete message, the CMTS tears down the RSVP link that uses the gate 37125.

RSVP-RESV-TEAR

Session-Object	Protocol	UDP	The protocol, Destination Address, Source Address, and Destination Port identify the RSVP flow.
	Destination Address	EMTAt	
	Destination port	7820	
Sender Templ	Source Address	MTAo	
	Source port	8422	

27) The EMTAo, upon receiving RSVP-RESV-TEAR, sends the RSVP-PATH-TEAR to CMTSo.

RSVP-PATH-TEAR

Session-Object	Protocol	UDP	These parameters identify the IP flow that is being terminated.
	Destination Address	MTAt	
	Destination port	7820	
Sender Templ	Source Address	MTAo	
	Source port	8422	

Appendix XV

Sample protocol message exchanges for Call Waiting using NCS

The following example (Figure XV.1) illustrates how Call Waiting is handled when using NCS signalling and CM initiated DSx messaging. The call flow below assumes that a call is already in progress between MTAo and MTA_{T1} using GateId #1 (37125) and ServiceFlows #1 (1001/2001). The 2nd connection for MTA_{T2} opens a new Gate (37130) and ServiceFlows (1002/2002) and uses the ResourceId (3333) returned from the initial call to indicate to the CMTS to share the underlying bandwidth for these 2 service flows.

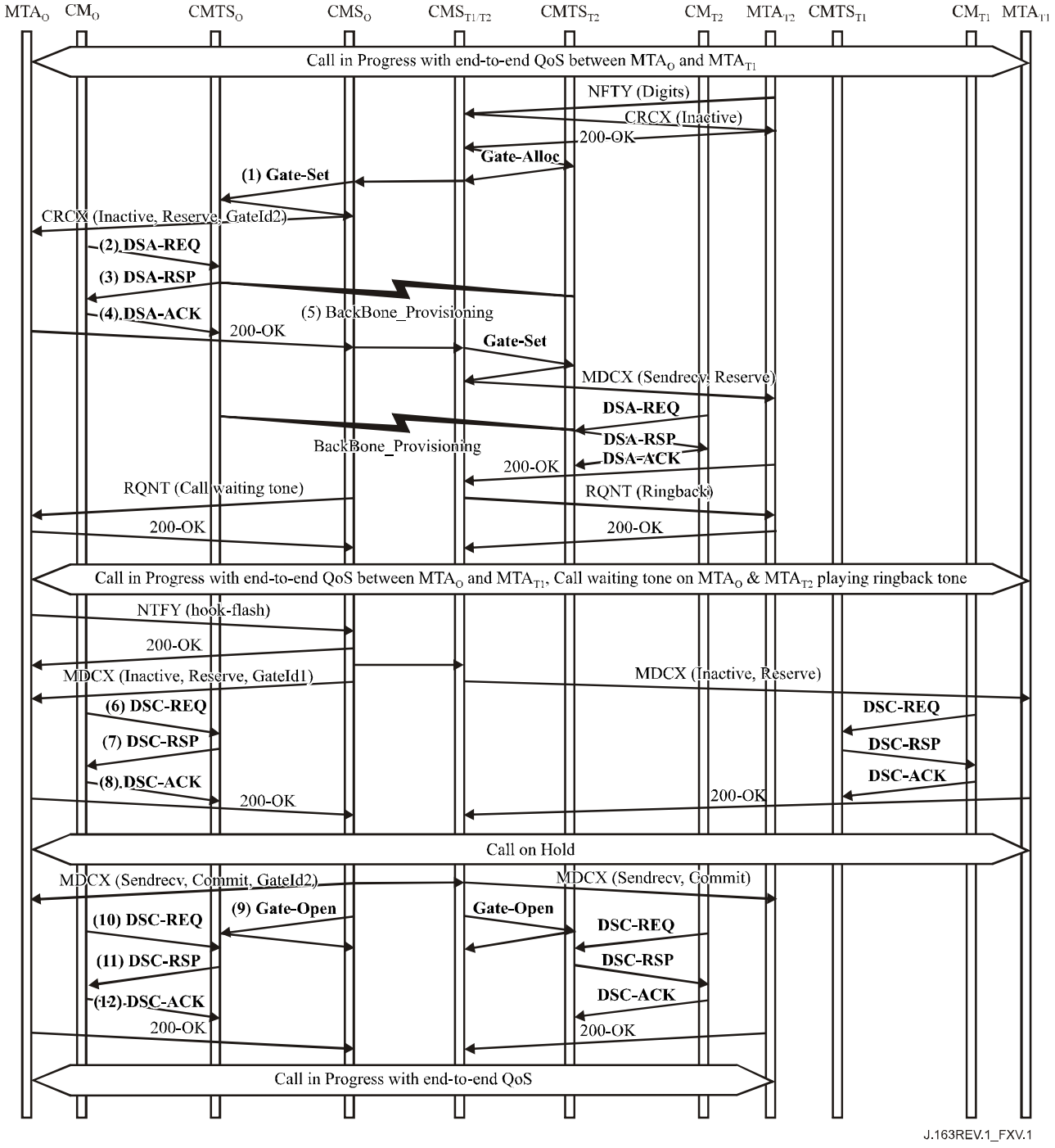


Figure XV.1/J.163 – Call Waiting using NCS

1) CMSo, upon receipt of signalling information from $CMS_{T1/T2}$, gives CMTSo authorization to admit the new connection.

GATE-SET

Transaction ID		3177	Unique Transaction ID for this message exchange.
Subscriber		MTAo	Request for total resources in use by this client.
Remote-Gate-Info	CMTS Address	CMSo	Information needed to perform gate coordination. Note that CMS has given itself as the entity for exchanging gate coordination messages.
	CMTS Port	2052	
	Remote Gate-ID	8095	
	Security Key	<key>	
	Flag	No-send-gate-open	
Event-Generation-Info	RKS-Addr	RKS	Address of Record Keeping Server.
	RKS-Port	3288	Port on Record Keeping Server.
	Billing Correlation ID	<id>	Opaque data that will be passed to RKS when resources are committed.
Gate-Spec	Direction	up	
	Protocol	UDP	The protocol, Destination Address, Source Address, and Destination Port quadruple are used for QoS classifiers.
	Source Address	MTAo	
	Destination Address	MTAt2	
	Source port	0	
	Destination port	7000	
	DSCP	6	Packet Type value for upstream packets.
	T1	180000	Maximum time between reserve and commit.
	T2	2000	Maximum time for gate coordination to complete.
	r	12000	These are the maximum bandwidth parameters that MTAo is authorized to request for this conversation.
	b	120	
	p	12000	
	m	120	
	M	120	
	R	12000	
	S	0	

GATE-SET

Gate-Spec	Direction	down	
	Protocol	UDP	The protocol, Destination Address, Source Address, and Destination Port quadruple are used for QoS classifiers.
	Source Address	MTAt2	
	Destination Address	MTAo	
	Source port	0	
	Destination port	7120	
	DSCP	9	Packet Type value for downstream packets.
	T1	180000	Maximum time between reserve and commit.
	T2	2000	Maximum time for gate coordination to complete.
	r	12000	These are the maximum bandwidth parameters that MTAo is authorized to request for this conversation.
	b	120	
	p	12000	
	m	120	
	M	120	
	R	12000	
	S	0	

CMTSo responds to the Gate Setup command with an acknowledgement.

GATE-SET-ACK

TransactionID		3177	
Subscriber		MTAo	Request for total resources in use by this client.
Gate-ID		37130	Identifier for allocated Gate.
Activity Count		3	Total connections established by this client.

2) MTAo, upon receiving a CRCX command from the CMS, calculates the QoS parameters for the DOCSIS 1.1 link. It uses the Annex E to Annex B/J.112 interface to the Cable Modem to send the following DSA-REQ to the CMTS. This message is used to establish both upstream and downstream parameters. The Upstream Unsolicited Grant Size was calculated as 120 (from SDP) plus 18 (Ethernet overhead) minus 40 (Header Suppression amount) plus 13 (DOCSIS overhead). Header suppression indicates the 42 bytes of Ethernet/IP/UDP header. Contents of the suppressed header is included in the DSA-REQ.

NOTE – This DSA identifies GateId2 (37130) and the ResourceId returned from the CRCX of the original call (3333) in the Authorization Block. This informs the CMTS that this service flow should share the underlying DOCSIS resources with the service flow for the original call.

DSA-REQ

TransactionID		1
UpstreamServiceFlow	ServiceFlowReference	1
	QoSParameterSetType	Admitted (2)
	TimeOutAdmitted	200
	ServiceFlowScheduling	UGS (6)
	NominalGrantInterval	10 ms
	ToleratedGrantJitter	2 ms
	GrantsPerInterval	1
	UnsolicitedGrantSize	111
DownstreamServiceFlow	ServiceFlowReference	2
	QoSParameterSetType	Admitted (2)
	TimeOutAdmitted	200
	TrafficPriority	5
	MaximumSustainedRate	12000
UpstreamPacketClassification	ServiceFlowReference	1
	PacketClassifierReference	1
	ClassifierPriority	150
	ClassifierActivationState	Inactive (0)
	IPSourceAddress	MTAo
	IPSourcePort	7120
	IPDestinationAddress	MGt2
	IPDestinationPort	7000
	IPProtocol	UDP (17)
DownstreamPacketClassification	ServiceFlowReference	2
	PacketClassifierReference	2
	ClassifierPriority	150
	ClassifierActivationState	Inactive (0)
	IPSourceAddress	MGt2
	IPDestinationAddress	MTAo
	IPDestinationPort	7124
	IPProtocol	UDP (17)
PayloadHeaderSuppression	ClassifierReference	1
	ServiceFlowReference	1
	HeaderSuppressionIndex	1
	HeaderSuppressionField	<42bytes>
	HeaderSuppressionMask	<42bits>
	HeaderSuppressionSize	42
	HeaderSuppressionVerify	Verify (0)
AuthorizationBlock	GateId	37130
	ResourceId	3333
HMAC		

3) The CMTS checks the authorization, by looking for a gate with gate-ID matching the value in AuthBlock, and checks the resources it is required to allocate (e.g., header suppression table space, Service Flow IDs, classifier table space), and installs the classifiers. If the operation is successful, it returns the DSA-RSP message stating the success.

DSA-RSP

TransactionID		1
ConfirmationCode		Success (0)
UpstreamServiceFlow	ServiceFlowReference	1
	ServiceFlowIdentifier	1002
	QoSParameterSetType	Admitted (2)
	TimeOutAdmitted	200
	ServiceFlowScheduling	UGS (6)
	NominalGrantInterval	10 ms
	ToleratedGrantJitter	2 ms
	GrantsPerInterval	1
	UnsolicitedGrantSize	111
DownstreamServiceFlow	ServiceFlowReference	2
	ServiceFlowIdentifier	2002
	QoSParameterSetType	Admitted (2)
	TimeOutAdmitted	200
	TrafficPriority	5
	MaximumSustainedRate	12000
UpstreamPacketClassification	ServiceFlowReference	1
	PacketClassifierReference	1
	PacketClassifierIdentifier	3001
	ClassifierPriority	150
	ClassifierActivationState	Inactive (0)
	IPSourceAddress	MTAo
	IPSourcePort	7120
	IPDestinationAddress	MGt2
	IPDestinationPort	7000
	IPProtocol	UDP (17)
DownstreamPacketClassification	ServiceFlowReference	2
	PacketClassifierReference	2
	PacketClassifierIdentifier	3002
	ClassifierPriority	150
	ClassifierActivationState	Inactive (0)
	IPSourceAddress	MGt2
	IPDestinationAddress	MTAo
	IPDestinationPort	7124
	IPProtocol	UDP (17)

DSA-RSP

AuthorizationBlock	ResourceId	3333
HMAC		

4) Upon receipt of the DSA-RSP, the CM acknowledges receipt with a DSA-ACK message.

DSA-ACK

TransactionID		1
ConfirmationCode		Success (0)
HMAC		

5) Simultaneous with message No. 3, the CMTS initiates any required backbone reservations for the requested quality of service. The content of this message is dependent on the particular backbone algorithms in use, and is outside the scope of this Recommendation. The backbone router sends to the CMTS any required notification that the reservation is successful.

6) In response to signalling messages that indicate the user wants to switch callers (i.e., a hook-flash is detected on MTAo), MTAo uses the Annex E to Annex B/J.112 interface to de-activate the admitted resources on ServiceFlow No. 1. This is done via a DSC-REQ DOCSIS 1.1 command to the CMTS.

DSC-REQ

TransactionID		2
UpstreamServiceFlow	ServiceFlowIdentifier	1001
	QoSParameterSetType	Admitted (2)
	TimeOutAdmitted	200
	ServiceFlowScheduling	UGS (6)
	NominalGrantInterval	10 ms
	ToleratedGrantJitter	2 ms
	GrantsPerInterval	1
	UnsolicitedGrantSize	111
DownstreamServiceFlow	ServiceFlowIdentifier	2001
	QoSParameterSetType	Admitted (2)
	TimeOutAdmitted	200
	TrafficPriority	5
	MaximumSustainedRate	12000
UpstreamPacketClassification	ServiceFlowIdentifier	1001
	PacketClassifierIdentifier	3001
	ClassifierChangeAction	Replace (1)
	ClassifierPriority	150
	ClassifierActivationState	Inactive (0)
	IPSourceAddress	MTAo
	IPSourcePort	7120
	IPDestinationAddress	MGt1
	IPDestinationPort	7000
	IPProtocol	UDP (17)

DSC-REQ

DownstreamPacketClassification	ServiceFlowIdentifier	2001
	PacketClassifierIdentifier	3002
	ClassifierChangeAction	Replace (1)
	ClassifierPriority	150
	ClassifierActivationState	Inactive (0)
	IPSourceAddress	MGt1
	IPDestinationAddress	MTAo
	IPDestinationPort	7124
	IPProtocol	UDP (17)
AuthorizationBlock	GateId	37125
HMAC		

7) The CMTS sends a DSC-RSP message showing the operation was successful.

DSC-RSP

TransactionID		2
ConfirmationCode		Success (0)
HMAC		

8) The CM sends DSC-ACK message to indicate that the DSC-RSP has been received and agreed.

DSC-ACK

TransactionID		2
ConfirmationCode		Success (0)
HMAC		

9) The CMS sends the gate open message to the CMTS to inform it that the resources for Gate No. 2 should be committed. If the CMTS does not receive the DSC-REQ from MTA within a short time, it should revoke the gate authorization.

GATE-OPEN

Transaction ID		72	Identifier to match this message with its response.
Gate ID		37130	Gate-ID at CMTS.
HMAC			Security checksum for this message.

The CMTS responds to the GATE-OPEN with:

GATE-OPEN-ACK

Transaction ID		72	Identifier to match this message with its response.
HMAC			Security checksum for this message.

10) After disabling the connection on ServiceFlow No. 1 the CMS signals the MTA to activate the resources on connection No. 2, MTAo uses the Annex E to Annex B/J.112 interface to activate the admitted resources. This is done via a DSC-REQ DOCSIS 1.1 command to the CMTS.

DSC-REQ

TransactionID		2
UpstreamServiceFlow	ServiceFlowIdentifier	1002
	QoSParameterSetType	Admitted + Activated (6)
	TimeOutActive	10
	ServiceFlowScheduling	UGS (6)
	NominalGrantInterval	10 ms
	ToleratedGrantJitter	2 ms
	GrantsPerInterval	1
	UnsolicitedGrantSize	111
DownstreamServiceFlow	ServiceFlowIdentifier	2002
	QoSParameterSetType	Admitted + Activated (6)
	TimeOutActive	10
	TrafficPriority	5
	MaximumSustainedRate	12000
UpstreamPacketClassification	ServiceFlowIdentifier	1002
	PacketClassifierIdentifier	3001
	ClassifierChangeAction	Replace (1)
	ClassifierPriority	150
	ClassifierActivationState	Active (1)
	IPSourceAddress	MTAo
	IPSourcePort	7120
	IPDestinationAddress	MGt2
	IPDestinationPort	7000
	IPProtocol	UDP (17)
DownstreamPacketClassification	ServiceFlowIdentifier	2002
	PacketClassifierIdentifier	3002
	ClassifierChangeAction	Replace (1)
	ClassifierPriority	150
	ClassifierActivationState	Active (1)
	IPSourceAddress	MGt2
	IPDestinationAddress	MTAo
	IPDestinationPort	7124
	IPProtocol	UDP (17)
AuthorizationBlock	GateId	37130
HMAC		

11) The CMTS sends a DSC-RSP message showing the operation was successful.

DSC-RSP

TransactionID		2
ConfirmationCode		Success (0)
HMAC		

12) The CM sends DSC-ACK message to indicate that the DSC-RSP has been received and agreed.

DSC-ACK

TransactionID		2
ConfirmationCode		Success (0)
HMAC		

SERIES OF ITU-T RECOMMENDATIONS

Series A	Organization of the work of ITU-T
Series B	Means of expression: definitions, symbols, classification
Series C	General telecommunication statistics
Series D	General tariff principles
Series E	Overall network operation, telephone service, service operation and human factors
Series F	Non-telephone telecommunication services
Series G	Transmission systems and media, digital systems and networks
Series H	Audiovisual and multimedia systems
Series I	Integrated services digital network
Series J	**Cable networks and transmission of television, sound programme and other multimedia signals**
Series K	Protection against interference
Series L	Construction, installation and protection of cables and other elements of outside plant
Series M	TMN and network maintenance: international transmission systems, telephone circuits, telegraphy, facsimile and leased circuits
Series N	Maintenance: international sound programme and television transmission circuits
Series O	Specifications of measuring equipment
Series P	Telephone transmission quality, telephone installations, local line networks
Series Q	Switching and signalling
Series R	Telegraph transmission
Series S	Telegraph services terminal equipment
Series T	Terminals for telematic services
Series U	Telegraph switching
Series V	Data communication over the telephone network
Series X	Data networks and open system communications
Series Y	Global information infrastructure, Internet protocol aspects and Next Generation Networks
Series Z	Languages and general software aspects for telecommunication systems

Printed in Switzerland
Geneva, 2004