INTERNATIONAL TELECOMMUNICATION UNION

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

H.225.0

(07/2003)

SERIES H: AUDIOVISUAL AND MULTIMEDIA SYSTEMS

Infrastructure of audiovisual services – Transmission multiplexing and synchronization

Call signalling protocols and media stream packetization for packet-based multimedia communication systems

ITU-T Recommendation H.225.0

ITU-T H-SERIES RECOMMENDATIONS

AUDIOVISUAL AND MULTIMEDIA SYSTEMS

CHARACTERISTICS OF VISUAL TELEPHONE SYSTEMS	H.100–H.199
INFRASTRUCTURE OF AUDIOVISUAL SERVICES	
General	H.200–H.219
Transmission multiplexing and synchronization	**H.220–H.229**
Systems aspects	H.230–H.239
Communication procedures	H.240–H.259
Coding of moving video	H.260–H.279
Related systems aspects	H.280–H.299
SYSTEMS AND TERMINAL EQUIPMENT FOR AUDIOVISUAL SERVICES	H.300–H.399
SUPPLEMENTARY SERVICES FOR MULTIMEDIA	H.450–H.499
MOBILITY AND COLLABORATION PROCEDURES	
Overview of Mobility and Collaboration, definitions, protocols and procedures	H.500–H.509
Mobility for H-Series multimedia systems and services	H.510–H.519
Mobile multimedia collaboration applications and services	H.520–H.529
Security for mobile multimedia systems and services	H.530–H.539
Security for mobile multimedia collaboration applications and services	H.540–H.549
Mobility interworking procedures	H.550–H.559
Mobile multimedia collaboration inter-working procedures	H.560–H.569
BROADBAND AND TRIPLE-PLAY MULTIMEDIA SERVICES	
Broadband multimedia services over VDSL	H.610–H.619

For further details, please refer to the list of ITU-T Recommendations.

ITU-T Recommendation H.225.0

Call signalling protocols and media stream packetization for packet-based multimedia communication systems

Summary

This Recommendation covers the technical requirements for narrow-band visual telephone services defined in H.200 and F.720-series Recommendations, in those situations where the transmission path includes one or more packet-based networks, each of which is configured and managed to provide a non-guaranteed Quality of Service (QoS) which is not equivalent to that of N-ISDN, such that additional protection or recovery mechanisms beyond those mandated by ITU-T Rec. H.320 need be provided in the terminals. It is noted that ITU-T Rec. H.322 addresses the use of some other LANs which are able to provide the underlying performance not assumed by the ITU-T Recs H.323 and H.225.0.

This Recommendation describes how audio, video, data, and control information on a packet-based network can be managed to provide conversational services in H.323 equipment.

Annex G describes methods to allow address resolution between administrative domains in H.323 systems for the purpose of completing calls between the administrative domains. An administrative domain exposes itself to other administrative domains through a type of logical element known as a border element.

Products claiming compliance with Version 5 of H.225.0 (this version) shall comply with all of the mandatory requirements of this Recommendation. Version 5 products can be identified by H.225.0 messages containing a **protocolIdentifier** value of {itu-t (0) recommendation (0) h (8) 2250 version (0) 5}.

Source

ITU-T Recommendation H.225.0 was approved by ITU-T Study Group 16 (2001-2004) under the ITU-T Recommendation A.8 procedure on 14 July 2003.

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 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

Page

Page

ITU-T Recommendation H.225.0

Call signalling protocols and media stream packetization for packet-based multimedia communication systems

The ITU-T,

considering

the widespread adoption of and the increasing use of ITU-T Rec. H.320 for videophony and videoconferencing services over networks conforming to the N-ISDN characteristics specified in the I-series Recommendations,

appreciating

the desirability and benefits of enabling the above services to be carried, wholly or in part, over Local Area Networks while also maintaining the capability of interworking with H.320 terminals,

and noting

the characteristics and performances of the many types of Local Area Network which are of potential interest,

recommends

that systems and equipment meeting the requirements of ITU-T Rec. H.322 or ITU-T Rec. H.323 are utilized to provide these facilities.

1 Scope

This Recommendation describes the means by which audio, video, data and control are associated, coded, and packetized for transport between H.323 equipment on a packet-based network. This includes the use of an H.323 gateway, which in turn may be connected to H.320, H.324, or H.310/H.321 terminals on N-ISDN, GSTN, or B-ISDN respectively. The equipment descriptions and procedures are described in ITU-T Rec. H.323 while this Recommendation covers protocols and message formats. Communication via an H.323 gateway to an H.322 gateway for guaranteed Quality of Service (QoS) LANs and thus to H.322 endpoints is also possible.

This Recommendation is intended to operate over a variety of different packet-based networks, including IEEE 802.3, Token Ring, etc. Thus, this Recommendation is defined as being above the Transport layer such as TCP/IP/UDP, SPX/IPX, etc. Specific profiles for particular transport protocol suites are included in Appendix IV. ***Thus, the scope of H.225.0 communication is between H.323 entities on the same packet-based network, using the same transport protocol***. This packet-based network may be a single segment or ring, or it logically could be an enterprise data network comprising multiple packet-based networks bridged or routed to create one interconnected network. It should be emphasized that operation of H.323 terminals over the entire Internet, or even several connected packet-based networks may result in poor performance. The possible means by which quality of service might be assured on this packet-based network, or on the Internet in general, is beyond the scope of this Recommendation. However, this Recommendation provides a means for the user of H.323 equipment to determine that quality problems are the result of packet-based network congestion, as well as procedures for corrective actions. It is also noted that the use of multiple H.323 gateways connected over the public ISDN network is a straightforward method for increasing quality of service.

ITU-T Rec. H.323 and this Recommendation are intended to extend H.320 and H.221 connections onto the non-guaranteed QoS packet-based network environment conferences. As such, the primary conference model[1] is one with size in the range of a few participants to a few thousand, as opposed to large-scale broadcast operations, with strong admission control, and tight conference control.

This Recommendation makes use of (RTP/RTCP) Real-time Transport Protocol/Real-Time Transport Control Protocol for media stream packetization and synchronization for all underlying packet-based networks (see Annexes A, B and C). Please note that the usage of RTP/RTCP as specified in this Recommendation is not tied in any way to the usage of TCP/IP/UDP. This Recommendation assumes a call model where initial signalling on a non-RTP transport address is used for call establishment and capability negotiation (see ITU-T Recs H.323 and H.245), followed by the establishment of one or more RTP/RTCP connections. This Recommendation contains details on the usage of RTP/RTCP.

In ITU-T Rec. H.221, audio, video, data, and control are multiplexed into one or more synchronized physical SCN calls. On the packet-based network side of an H.323 call, none of these concepts apply. There is no need to carry from the SCN side the H.221 concept of a P × 64 kbit/s call, e.g., 2 by 64 kbit/s, 3 by 64 kbit/s, etc. Thus, on the packet-based network side, for example, there are only single "connection" calls with a maximum rate limited to 128 kbit/s, not 2 × 64 kbit/s fixed rate calls. Another example has single "connection" packet-based network calls with a maximum rate limited to 384 kbit/s interworking with 6 × 64 kbit/s on the SCN side[2]. The primary rationale of this approach is to put complexity in the gateway rather than the terminal and to avoid extending onto the packet-based network features of H.320 that are tightly tied to ISDN unless this is necessary.

In general, H.323 terminals are not aware directly of the H.320 transfer rate while interworking through an H.323 gateway; instead, the gateway uses H.245 **FlowControlCommand** messages to limit the media rate on each logical channel in use to that allowed by the H.221 multiplex. The gateway may allow the packet-based network side video rates to substantially underrun the SCN side rates (or the reverse) though the usage of a rate reducing function and H.261 fill frames; the details of such operations are beyond the scope of ITU-T Rec. H.323 and this Recommendation. Note that the H.323 terminal is indirectly aware of the H.320 transfer rates via the video maximum bit rate fields in ITU-T Rec. H.245 and shall not transmit at rates that exceed these rates.

This Recommendation is designed so that, with an H.323 gateway, interoperability with H.320 (1990), H.320 (1993), and H.320 (1996) terminals is possible. However, some features of this Recommendation may be directed toward allowing enhanced operations with future versions of ITU-T Rec. H.320. It is also possible that the quality of service on the H.320 side may vary based on the features and capabilities of the H.323 gateway (see Figure 1).

[1] An optional broadcast-only conference model is under consideration; of necessity, the broadcast model does not provide tight admissions or conference control.

[2] Note that video and data rates on the LAN side must match the video and data rates in the SCN side H.320 multiplex; the audio and control rates are not required to match. Stated another way, one would normally expect that, using H.245 flow control, the LAN/SCN gateway will force the video and data rates to fit into the H.221 SCN multiplex. However, since audio may be transcoded in the gateway often, one will frequently find that the LAN audio rate and the SCN rate do not match. Also, there should be no expectation that the H.221 bit rate for control (800 bit/s) will generally match the H.245 bit rate on the LAN side. Also note that the LAN rate may under-run the SCN rate for either/both video or/and data, but it cannot exceed the maximum amount that fits into the SCN side multiplex.

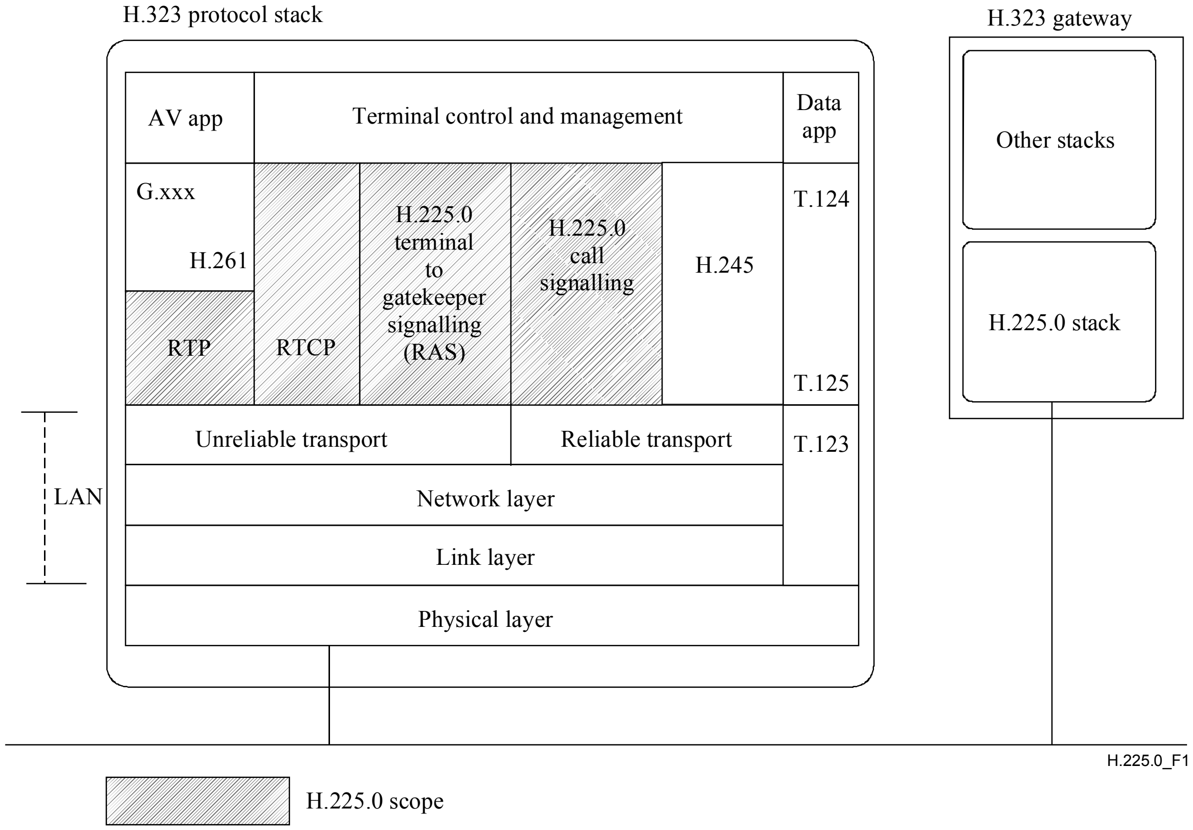


Figure 1/H.225.0 – H.225.0 scope

The general approach of this Recommendation is to provide a means of synchronizing packets that make use of the underlying packet-based network/transport facilities. This Recommendation does not require all media and control to be mixed into a single stream, which is then packetized. The framing mechanisms of ITU-T Rec. H.221 are not utilized for the following reasons:

- Not using H.221 allows each media to receive different error treatment as appropriate.
- H.221 is relatively sensitive to the loss of random groups of bits; packetization allows greater robustness in the packet-based network environment.
- H.245 and H.225.0 call signalling messsages can be sent over reliable links provided by the packet-based network.
- The flexibility and power of H.245 as compared to H.242.

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.

[1] ITU-T Recommendation G.711 (1988), *Pulse code modulation (PCM) of voice frequencies*.

[2] ITU-T Recommendation G.722 (1988), *7 kHz audio-coding within 64 kbit/s*.

[3] ITU-T Recommendation G.728 (1992), *Coding of speech at 16 kbit/s using low-delay code excited linear prediction*.

[4] ITU-T Recommendation G.723.1 (1996), *Speech coders: Dual rate speech coder for multimedia communications transmitting at 5.3 and 6.3 kbit/s*.

[5] ITU-T Recommendation G.729 (1996), *Coding of speech at 8 kbit/s using conjugate-structure algebraic-code-excited linear prediction (CS-ACELP)*.

[6] ITU-T Recommendation H.221 (1999), *Frame structure for a 64 to 1920 kbit/s channel in audiovisual teleservices*.

[7] ITU-T Recommendation H.230 (1999), *Frame-synchronous control and indication signals for audiovisual systems*.

[8] ITU-T Recommendation H.233 (1995), *Confidentiality system for audiovisual services*.

[9] ITU-T Recommendation H.242 (1999), *System for establishing communication between audiovisual terminals using digital channels up to 2 Mbit/s*.

[10] ITU-T Recommendation H.243 (2000), *Procedures for establishing communication between three or more audiovisual terminals using digital channels up to 1920 kbit/s*.

[11] ITU-T Recommendation H.245 (2003), *Control protocol for multimedia communication*.

[12] ITU-T Recommendation H.261 (1993), *Video codec for audiovisual services at $p \times 64$ kbit/s*.

[13] ITU-T Recommendation H.263 (1998), *Video coding for low bit rate communication*.

[14] ITU-T Recommendation H.320 (1999), *Narrow-band visual telephone systems and terminal equipment*.

[15] ITU-T Recommendation T.122 (1998), *Multipoint communication service – Service definition*.

[16] ITU-T Recommendation T.123 (1999), *Network-specific data protocol stacks for multimedia conferencing*.

[17] ITU-T Recommendation T.125 (1998), *Multipoint communication service protocol specification*.

[18] ITU-T Recommendation H.321 (1998), *Adaptation of H.320 visual telephone terminals to B-ISDN environments*.

[19] ITU-T Recommendation H.322 (1996), *Visual telephone systems and terminal equipment for local area networks which provide a guaranteed quality of service*.

[20] ITU-T Recommendation H.324 (1998), *Terminal for low bit-rate multimedia communication*.

[21] ITU-T Recommendation H.310 (1998), *Broadband audiovisual communication systems and terminals*.

[22] ITU-T Recommendation Q.931 (1998), *ISDN user-network interface layer 3 specification for basic call control*.

[23] ITU-T Recommendation Q.932 (1998), *Digital subscriber signalling system No. 1 – Generic procedures for the control of ISDN supplementary services*.

[24] ITU-T Recommendation X.680 (2002) | ISO/IEC 8824-1:2002, *Information technology – Abstract Syntax Notation One (ASN.1): Specification of basic notation*.

[25] ITU-T Recommendation X.681/(2002) | ISO/IEC 8824-2:2002, *Information technology – Abstract Syntax Notation One (ASN.1): Information object specification*.

[26] ITU-T Recommendation X.691 (2002) | ISO/IEC 8825-2:2002, *Information technology – ASN.1 encoding rules: Specification of Packed Encoding Rules (PER)*.

[27] ITU-T Recommendation E.164 (1997), *The international public telecommunication numbering plan*.

[28] ISO/IEC 10646-1:2000, *Information technology – Universal Multiple-Octet Coded Character Set (UCS) – Part 1: Architecture and Basic Multilingual Plane*.

[29] ITU-T Recommendation Q.850 (1998), *Usage of cause and location in the digital subscriber Signalling System No. 1 and the Signalling System No. 7 ISDN user part*.

[30] ITU-T Recommendation Q.950 (2000), *Supplementary services protocols, structure, and general principles*.

[31] ITU-T Recommendation H.235 (2000), *Security and encryption for H-series (H.323 and other H.245-based) multimedia terminals*.

[32] ISO/IEC 11571:1998, *Information technology – Telecommunications and information exchange between systems – Private Integrated Services Networks – Addressing*.

[33] IETF RFC 1738 (1994), *Uniform Resource Locators (URL)*.

[34] IETF RFC 2068 (1997), *Hypertext Transfer Protocol – HTTP/1.1*.

[35] IETF RFC 1766 (1995), *Tags for the Identification of Languages*.

[36] ITU-T Recommendation H.248 (2000), *Gateway control protocol*.

[37] IETF RFC 3550 (2003), *RTP: A Transport Protocol for Real-Time Applications*.

[38] IETF RFC 3551 (2003), *RTP Profile for Audio and Video Conferences with Minimal Control*.

[39] IETF RFC 2032 (1996), *RTP Payload Format for H.261 Video Streams*.

[40] ITU-T Recommendation X.690 (2002) | ISO/IEC 8825-1:2002, *Information technology – ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)*.

3 Definitions

See definitions in ITU-T Rec. H.323. In ITU-T Rec. H.323 the term "endpoint" is used to refer to terminals, gateways, and MCUs as elements that are capable of receiving or initiating calls. In this Recommendation the term "terminal" is often used in a general way in descriptions of call setup, and should be understood as referring to an element that can take part in call setup, including a gateway or MCU.

4 Conventions

In this Recommendation, "shall" refers to a mandatory requirement, while "should" refers to a suggested but optional feature or procedure. The term "may" refers to an optional course of action without expressing a preference.

When a term such as "MCU" is used, an H.323 MCU is referred to. If an H.231 MCU is intended, this will be explicitly noted.

In this Recommendation kilobits/second is abbreviated kbit/s and is measured in units of 1000. Thus, 64 kbit/s is exactly 64 000 bits per second.

Unless otherwise specified, the aligned variant PER encoding of ASN.1 shall be used for all ASN.1 specified in this Recommendation.

Q.931 message names are Capitalized. ASN.1 is in **bold**.

5 Abbreviations

This Recommendation uses the following abbreviations:

5.1 General abbreviations

BAS	Bit rate Allocation Signal
CIF	Common Intermediate Format
CRV	Call Reference Value
ECS	Encryption Control Signal
FFS	For Further Study
GOB	Group of Blocks
H-MLP	High speed Multi-Layer Protocol
HSD	High Speed Data
IA5	International Alphabet No. 5
IE	Information Element
IETF	Internet Engineering Task Force
IP	Internet Protocol
LAN	Local Area Network
LD-CELP	Low Delay – Code Excited Linear Prediction
LSB	Least Significant Bit
LSD	Low Speed Data
MB	Macro Block (see ITU-T Rec. H.261)
MBE	Multi-Byte Extension
MCC	Multipoint Command Conference
MCN	Multipoint Command Negating
MCS	Multipoint Command Symmetrical data transmission
MCS	Multipoint Communication Service
MCU	Multipoint Control Unit
MF	MultiFrame
MLP	Multi-Layer Protocol
MPI	Minimum Picture Interval
MSB	Most Significant Bit
NA	Not Applicable
NS	Non-Standard
NSAP	Network Service Access Point
PCM	Pulse Code Modulation
PDU	Protocol Data Unit
QCIF	Quarter Common Intermediate Format
QoS	Quality of Service

RAS	Registration, Admission and Status
RTCP	Real-time Transport Control Protocol
RTP	Real-time Transport Protocol
SBE	Single Byte Extension
SC	Service Channel
SCM	Selected Communication Mode
SCN	Switched Circuit Network
TCP	Transport Control Protocol
TSAP	Transport Service Access Point
UDP	User Datagram Protocol
URL	Uniform Resource Locator
VCF	Video Command "Freeze picture Request"
VCU	Video Command "Fast Update Request"

5.2 RAS message abbreviations

ACF	Admissions Confirm
ARJ	Admissions Reject
ARQ	Admissions Request
BCF	Bandwidth Confirm
BRJ	Bandwidth Reject
BRQ	Bandwidth Request
DCF	Disengage Confirm
DRJ	Disengage Reject
DRQ	Disengage Request
GCF	Gatekeeper Confirm
GRJ	Gatekeeper Reject
GRQ	Gatekeeper Request
IACK	Information request Acknowledgement
INAK	Information request Negative Acknowledgement
IRQ	Information Request
IRR	Information Request Response
LCF	Location Confirm
LRJ	Location Reject
LRQ	Location Request
RAC	Resource Availability Confirmation
RAI	Resource Availability Indication
RCF	Registration Confirm
RIP	Request In Progress

RRJ	Registration Reject
RRQ	Registration Request
SCI	Service Control Indication
SCR	Service Control Response
UCF	Unregistration Confirm
URJ	Unregistration Reject
URQ	Unregistration Request

6 Packetization and synchronization mechanism

6.1 General approach

Before any calls are made, an endpoint may discover/register with a gatekeeper. If this is the case, it is desirable for the endpoint to know the vintage of the gatekeeper it is registering with. It is also desirable for the gatekeeper to know the vintage of endpoints that register with it. For these reasons, both the *discovery* and registration sequences contain an H.245 style OBJECT IDENTIFIER that allows the vintage to be determined in terms of the version of ITU-T Rec. H.323 implemented. This sequence also may contain optional non-standard message parts to allow endpoints to establish non-standard relationships. At the end of this sequence, both gatekeepers and endpoints are aware of the version numbers and the non-standard status of each other.

The version number is mandatory and non-standard information is optional in the Setup/Connect sequence described below to allow two endpoints to inform each other of their vintage and non-standard status. Note, however, that all H.225.0 call signalling messages have a field for an optional non-standard message in the User-user information element, and that all RAS channel messages have an optional field for non-standard information. In addition, a non-standard RAS message has been defined that can be sent at any time.

The unreliable channel for registration, admissions, and status messaging is called the RAS channel. The general approach to starting a call is to send a mandatory admission request on the RAS channel[3], followed by an initial Setup message on a reliable channel transport address (this address may have been returned in the admission confirmation message, or may have been known to the calling terminal). As a result of this initial message, a call setup sequence commences based on H.225.0 call signalling operations with enhancements described below. The sequence is complete when the terminal receives in the Connect message, a reliable transport address on which to send H.245 control messages[4].

When messages are sent on the reliable H.225.0 call signalling channel, only one whole message shall be sent within the boundaries defined by the reliable transport; there shall be no fragmentation of H.225.0 messages across transport PDUs. (In IP implementations as outlined in Appendix IV, this PDU is defined by TPKT.)

Once the reliable H.245 control channel has been established, additional channels for audio, video, and data may be established based on the outcome of the capability exchange using H.245 logical channel procedures. Also, the nature of the packet-based network-side multi-media conference

[3] A terminal that is not registered with a gatekeeper is not required to send an admissions request.

[4] Note that the H.245 address may be sent in the Alerting or Call Proceeding message to shorten call setup time. Note that the H.245 channel may be opened immediately after the receipt of the H.245 address in the Setup message.

(centralized vs. distributed/multicast) is negotiated on a per connection basis[5]. This negotiation is performed per media, in the sense that, for example, audio/video may be distributed, while data and control are centralized.

When messages are sent on the reliable H.245 control channel, more than one message may be sent within the boundaries defined by the reliable transport PDU as long as whole messages are sent; there shall be no fragmentation of H.245 messages across transport PDUs. (In IP implementations as outlined in Appendix IV, this PDU is defined by TPKT.)

H.225.0 terminals shall be capable of sending audio and video using RTP via unreliable channels to minimize delay. Error concealment or other recovery action may be applied to overcome lost packets; in general, audio/video packets are not retransmitted since this would result in excessive delay in the packet-based network environment[6]. It is assumed that bit errors are detected in the lower layers, and errored packets are not sent up to H.225.0. Note that audio/video and call signalling/H.245 control are never sent on the same channel, and do not share a common message structure. H.225.0 terminals shall be capable of sending and receiving audio and video on separate transport addresses using separate instances of RTP to allow for media-specific frame sequence numbers and separate quality of service treatment for each media. However, an optional mode, where audio and video packets are mixed in a single frame which is sent to a single transport address, is for further study.

T.120 capabilities are negotiated using H.245, and upon receipt of appropriate messages, T.120 conferences are established using the transport/packet-based network stacks of T.123 as appropriate. T.120 shall be conveyed over the packet-based network between endpoints on another transport address. Table 1 shows the number of TSAP identifiers used for each media on a point-to-point call. It is also true that a given H.323 terminal may be able to participate in more than one conference at a time, resulting in the use of additional TSAP identifiers. All H.245 logical channels used are unidirectional except for those associated with T.120, which are bidirectional.

Table 1/H.225.0 – TSAP IDs used by H.225.0 per point-to-point unicast call

Usage of TSAP IDs	Reliable or unreliable	Well known or dynamic
Audio/RTP	Unreliable	Dynamic
Audio/RTCP	Unreliable	Dynamic
Video/RTP	Unreliable	Dynamic
Video/RTCP	Unreliable	Dynamic
Call Signalling	Reliable	Well known or dynamic
H.245	Reliable	Dynamic
Data (T.120)	Reliable	Well known or dynamic
RAS	Unreliable	Well known or dynamic
NOTE – If well known TSAP identifiers are used, there can only be a single endpoint per network address. Also, in the direct call model the caller requires a well known TSAP identifier for the Call Signalling channel to start the call.		

[5] The LAN side conference may be part centralized and part distributed, as decided by the MC controlling the conference. However, the terminal is not aware of this fact. Generally, of course, all terminals will see the same Selected Communications Mode (SCM) (see ITU-T Rec. H.243 for a definition).

[6] Fast Update of full frames, MBs, or GOBs may be requested via H.245 signalling.

Although the transport address for, say, audio and video, may share the same packet-based network address and differ only by TSAP identifier, some manufacturers may choose to use different packet-based network addresses for audio and video. The only requirement is that the convention of Annexes A and B should be followed in the numbering of TSAP identifiers in the RTP session[7].

Table 1 describes the basic case of point-to-point unicast operations between two terminals. To facilitate the construction of gateways, MCUs, and gatekeepers, dynamic TSAP IDs may be used instead of well known TSAP IDs. Tables 2 and 3 illustrate an example of TSAP ID usage for the gateway/MCU case, and for the gatekeeper case.

Table 2/H.225.0 – TSAP IDs used on one MCU/gateway port (Unicast example)

Usage of TSAP IDs	Reliable or unreliable	Well known or dynamic
Audio/RTP	Unreliable	Dynamic
Audio/RTCP	Unreliable	Dynamic
Video/RTP	Unreliable	Dynamic
Video/RTCP	Unreliable	Dynamic
Call Signalling	Reliable	Dynamic (Note)
H.245	Reliable	Dynamic
Data (T.120)	Reliable	Dynamic
RAS	Unreliable	Dynamic (Note)
NOTE – See Note 1 of Table 3.		

Table 3/H.225.0 – Example of TSAP IDs usage by H.225.0 gatekeeper supporting the gatekeeper mediated call model of Figure 28/H.323 for a point-to-point call

Usage of TSAP IDs	Reliable or unreliable	Well known or dynamic	Number of channels
Call Signalling	Reliable	Dynamic or well known (Note 1)	2 per call (Note 2)
H.245	Reliable	Dynamic	2 per call (Note 2)
RAS	Unreliable	Well known	1
NOTE 1 – If the well known TSAP ID is used the gatekeeper may be limited to a single endpoint per device; therefore, dynamic TSAP IDs should be used. NOTE 2 – 0 for direct call model; 2 for gatekeeper-mediated call model.			

Note that a reliable transport address is used for call setup for the terminal to terminal case, and also for the gatekeeper-mediated case. The reliable call signalling connection shall be kept active until a Release Complete message is received for all active calls signalled over the call signalling channel.

Note that more than one H.245 channel may be open at a given time, i.e., an endpoint may be in more than one call/conference at the same time. Note also that within a specific call, a terminal may have more than one channel of the same type open, e.g., two audio channels for stereo audio. The only limitation is that there shall be one and only one H.245 control channel per point-to-point call.

[7] Note that any TSAP ID can be used for the initial RTP session; the major reason to follow the RTP convention is for possible IETF RTP interoperability.

H.245 logical channel signalling is used to start and stop video, audio, and data protocol usage. This process calls for closing the open channel, and then reopening with a new mode of operation. As part of the process of opening the channel, before sending the open logical channel acknowledgment, the endpoint uses the ARQ/ACF or BRQ/BCF sequence to ensure that sufficient bandwidth is available for the new channel (unless sufficient bandwidth is available from a previous ARQ/ACF or BRQ/BCF sequence). In some cases, the gateway may find that the SCN side mode change occurs more quickly than the packet-based network side mode change, resulting in the possibility of the loss of audio information. The gateway may adopt several approaches at the discretion of the manufacturer:

a) the gateway may transcode audio, thus hiding the SCN mode changes;

b) the gateway may simply throw away audio information; or

c) the gateway may operate as an H.231 MCU, thus gaining control over all SCN side mode changes.

No general rule exists concerning whether H.245 or RTP procedures (see Annexes A, B and C) take precedence; each conflict and its resolution is specifically mentioned in this Recommendation.

Note also that there is no fixed association between SSRCs and logical channels; ITU-T Rec. H.245 provides this association which may be used for audio/video synchronization.

In general, two types of conference modes of operation on the packet-based network side are possible: distributed and centralized. It is also possible that different choices may be made for different media, e.g., distributed audio/video and centralized data. Procedures for determining what sort of conference to establish are in ITU-T Rec. H.323; the messages of this Recommendation are intended to support all allowed combinations, noting that distributed control and data are for further study although supported by the H.245 capability signalling.

6.2 Use of RTP/RTCP

The H.225.0 endpoint shall be capable of using separate TSAP IDs for audio and video and the associated RTCP channels as described in Annexes A and B. Optionally, endpoints may choose to use different packet-based network addresses for audio and video, but for each packet-based network address the convention of Annexes A and B should be followed in the use of TSAP IDs. Using H.245 signalling, additional audio and video channels may be established if the terminal supports this capability.

An optional capability to use a single transport address for both audio and video is for further study.

Unless an exception is specifically mentioned here, implementations shall follow those of RTP as contained in Annex A unless modified by text in this Recommendation. Implementations shall follow the RTP profile (see Annex B) only as specifically mentioned in this Recommendation.

RTP translators and mixers are not elements of the H.323 system, and any information about them in Annexes A and B shall be regarded as informative. Note that both gateways and MCUs have some aspects of both mixers and translators, and the information in Annexes A and B may be helpful in the implementation of gateways and MCUs. However, MCUs are not mixers, and mixers are not MCUs. Note that gateways, for example, on a packet-based network-to-packet-based network call via the gateway, may act as translators.

Version (V): Version 2 of RTP shall be used.

CSRC Count (CC): Use of the CSRC count in this Recommendation is optional. When not in use, the value of CC shall be zero (0). The CSRC may be used by MCUs to provide information on contributors to the audio sum when distributed audio processing is occurring. Note that there are no capabilities associated with the ability to understand the CSRC count so the MCU/MC has no way of knowing whether and how the terminal in the conference makes use of the information.

CNAME: In the simplest case of a point-to-point connection on the packet-based network, the SSRC is used to identify an audio/video source from a terminal, and the streams are associated by a CNAME as being supplied by the same endpoint as specified in Annex A.

When using RTCP, either RR or SR packets shall be sent periodically as described in Annex A. The CNAME SDES message shall be used. Other SDES messages (see Annex A) are optional, but shall not be used for conference control or conference information when either H.245 and/or T.120 control functions are in use. Information provided by ITU-T Rec. H.245 and/or ITU-T Rec. T.120 shall be regarded as the correct information.

The RTCP BYE message shall not be relied on for RTP session termination. The H.323 terminal determines when a call is disconnected via the procedures of ITU-T Rec. H.323. The only mandatory use of the RTCP BYE packet is for SSRC collision resolution.

The H.323 terminal, when engaged in any conference, whether point-to-point or multi-point, shall restrict the logical channel bit rate averaged over a period as defined in ITU-T Rec. H.245 to that signalled in the H.245 **FlowControlCommands**, H.245 logical channel commands, and the T.120 flow control mechanism.

When the H.323 terminal is connected to an H.323 gateway, the gateway shall use the means of ITU-T Rec. H.245 and ITU-T Rec. T.120 to force the H.323 terminal to transmit at a rate less than or equal to the SCN side media rates and receive at a rate equal or higher than the SCN rate, with the following exceptions:

- Control bandwidth on the packet-based network need not match that in ITU-T Rec. H.221.
- Audio bandwidth on the packet-based network may match that in ITU-T Rec. H.221 on the SCN, but with gateway transcoding a match is not required.
- In the case where the gateway is using a rate reducer: the packet-based network-side H.323 terminal shall not exceed the H.245 signalled rate, which will probably be less than the rate being sent over the SCN.

Encryption for H.323 endpoints is for further study.

6.2.1 Audio

Before considering how audio is packetized using RTP, attention must be directed toward how it is signalled via H.245, and the relationship of this signalling to RTP. In general, when the audio channel is opened, an H.245 logical channel is opened. H.245 signalling in the **AudioCapability** structure is given in terms of the maximum number of frames per packet. The frame size for this Recommendation varies with the audio coding in use.

All H.323 terminals offering audio communication shall support G.711. For all frame-oriented audio codecs, receivers shall signal the maximum number of audio frames they are capable of accepting in a single audio packet. Transmitters may send any whole number of audio frames in each packet, up to the maximum stated by the receiver. Transmitters shall not split audio frames across packets, and shall send whole numbers of octets in each audio packet.

Sample-based codecs, such as G.711 and G.722, shall be considered to be frame-oriented, with a frame size of eight samples. (See Annex B for additional information regarding guidelines for sample-based audio encoding.) For audio algorithms, such as G.723.1, which use more than one size of audio frame, audio frame boundaries within each packet shall be signalled in-band to the audio channel.

For audio algorithms which use a fixed frame size (see ITU-T Recs G.728 and G.729 for the frame size used by each), audio frame boundaries shall be implied by the ratio of packet size to audio frame size; in other words, only whole audio frames shall be put in the RTP packet.

Payload Type (PT): Only ITU-T payload types such as (0)[PCMU], (8)[PCMA], (9)[G722], and (15)[G728] shall be used for ITU codecs signalled in ITU-T Rec. H.245. Dynamic payload types exchanged using H.245 signalling shall be used for any ITU-T payload types not listed in Annex B.

It is recommended that if an interruption in sequence numbers is observed, the receiver may repeat the most recent received sounds such that the amplitude of the repeated sound decays to silence; other similar procedures may be used at the discretion of the manufacturer.

Each G.711 octet shall be octet aligned in an RTP packet. The sign bit of each G.711 octet shall correspond to the most significant bit of the octet in the RTP packet (i.e., assuming the G.711 samples are handled as octets on the host machine, the sign bit shall be the most significant bit of the octet as defined by the host machine format).

When sending 48/56 kbit/s PCM toward the packet-based network, the H.323 gateway shall pad the extra 1 or 2 bits in each octet in accordance with Note 2 in Table 1b/G.711, and use the RTP values for PCMA or PCMU(8 or 0). For µ-law the padding consists of "1" in both the 7th and 8th bit. For A-law the 7th bit shall be 0 and the 8th bit 1. In the reverse direction the H.323 gateway shall truncate 64 kbit/s G.711 on the packet-based network side to fit the G.711 rate being used in H.320. Thus, on the packet-based network side only 64 kbit/s G.711 shall be used.

When sending 48/56 kbit/s G.722 toward the packet-based network, the H.323 gateway shall pad the extra 1 or 2 bits in each octet, and use dynamic RTP payload types as signalled by ITU-T Rec. H.245 to differentiate between 64 kbit/s (which uses PT = 9) and the reduced rate cases. In the reverse direction, the H.323 gateway shall truncate 64 kbit/s G.722 on the packet-based network side to fit the G.711 rate being used in H.320. Thus, on the packet-based network side only 64 kbit/s G.722 shall be used.

If possible, the H.323 terminal should make use of the silence suppression feature of RTP, especially when the conference is multicast. The H.323 terminal shall be able to receive silence compressed RTP streams. Coders may omit sending audio signals during silent periods after sending a single frame of silence, or may send silence background fill frames if such techniques are specified by the audio codec Recommendation in use.

6.2.2 Video messages

Payload Type (PT): Only ITU-T payload types such as that for ITU-T Rec. H.261 or ITU-T Rec. H.263 shall be used for ITU codecs signalled in ITU-T Rec. H.245. Dynamic payload types may be used for codecs which can be signalled via H.245 and for which packetization formats have not been defined.

Marker (M): The marker bit should be set according to the procedures described in Annex A, except in cases where it would increase end-to-end delay.

In order to recover from the loss of video packets, H.245 **VideoFastUpdatePicture**, **VideoFastUpdateMB**, and **VideoFastUpdateGOB** shall be supported. Use of the RTCP control packets Full Intra Request (FIR) [send me a full frame] and Negative Acknowledgment (NACK) [Send me certain packets] is optional, and signalled in H.245 capabilities.

In RFC 2032 [39] section 5, error recovery method 3) may be impractical if the NACK does not arrive within one frame time.

H.261 is packetized on the packet-based network side as per Annex C. As long as sufficiently large RTP packets are available, fragmentation on MB boundaries by the transmitter is not required. However, if the H.323 terminal fragments H.261 packets on the RTP level, this fragmentation shall occur on MB boundaries. All H.323 terminals shall be able to receive MB fragmented packets as well as GOB fragmented packets, or packets with a mix of MBs and GOBs. Note that failure to support MB fragmentation in the transmitter may result in the loss of an entire GOB, and may also lower the packet rate. RTP packets used shall not exceed the size of the Maximum Transfer Unit (MTU) on a given packet-based network to maximize robustness of operation, but if the smallest independently coded element of the coding scheme (e.g., a macroblock) is larger than the MTU size it is not required to break up the packet over MTUs. MBs shall not be split across packets; all packets shall end on a GOB or MB boundary. The H.323 transmitter may choose to fill out a packet containing a small GOB with additional MBs, but this is not required.

To preclude the possibility of corruption in multiple pictures caused by the loss of an RTP packet, the RTP packetizer in an H.323 endpoint shall not include video from more than one picture in an RTP packet.

The SBIT is the number of most significant bits that shall be ignored in the first data octet. EBIT is the number of least significant bits that shall be ignored in the last data octet.

The RTP packetizer shall not intentionally octet align video at the start of RTP packets. In other words, if EBIT = n in an RTP packet, SBIT in the next RTP packet shall equal 8 – n, 0 < n < 8, and if EBIT = 0 in an RTP packet, SBIT in the next RTP packet shall equal 0. This requirement avoids possible additional end-to-end delay caused by bit-shifting. This requirement shall apply across picture boundaries.

Annex D specifies an H.323 extension to the video packet header that contains an optional octet count. The use of this optional extension is described in Annex D.

See Appendix IV for packet-based network-specific advice on video packetization.

6.2.3 Data messages

There are no special data messages or formats; T.120 is used on the packet-based network as per ITU-T Rec. T.123. Centralized vs. distributed data conferencing on the packet-based network is described in ITU-T Rec. H.323, and is negotiated via H.245.

T.120 flow control on the packet-based network is managed using packet-based network protocols when requested by H.245 **FlowControlCommand** and **maxBitRate** limits.

See ITU-T Rec. H.323 for the procedures used to connect a running T.120 conference to an H.323 conference, or to add an H.323 call to a T.120 conference.

The protocol to be used by H.224 on the packet-based network is for further study.

7 H.225.0 message definitions

This clause concerns the definition of messages for call setup, call control, and communications between terminals, gateways, gatekeepers, and MCUs.

The ASN.1 definitions for all H.225.0 messages appear in Annex H.

7.1 Use of Q.931 messages

Implementations shall follow ITU-T Rec. Q.931 as specified in this Recommendation. Terminals may also support optional H.450 APDUs in the User-User IE. The messages shall contain all of the mandatory information elements and may contain any of the optional information elements as defined in ITU-T Rec. Q.931 as described in this Recommendation. Note that the H.225.0 endpoint may, according to ITU-T Rec. Q.931, ignore all optional messages it does not support without harming interoperability, but shall respond to an unknown message with a Status message.

Each H.225.0 endpoint shall be able to receive and identify an incoming H.225.0 call signalling message including one containing an H.450 APDU in the User-user IE. It shall be capable of processing the mandatory H.225.0 call signalling messages; it may be capable of processing the optional H.225.0 call signalling messages. In any case, each H.225.0 endpoint shall be able to ignore messages unknown to it without disturbing operation.

Each H.225.0 endpoint shall be able to interpret and generate the information elements mandated in the following for the respective H.225.0 call signalling messages and H.450 APDU in User-user IE.

It may interpret and generate the optional information elements as defined below as well. It may also interpret other information elements of Q.931, or other Q-series or H.450 protocols. The endpoints shall be able to ignore unknown information elements contained in an H.225.0 call signalling message or H.450 APDU without disturbing operation. Procedures for receiving unrecognized "comprehension required" information elements shall apply according to 5.8.7.1/Q.931. H.225.0 endpoints shall not send multiple information elements of the same type in the same message; for example, they shall not send multiple Calling Party Number information elements as described in Annex A/Q.951.3.

Information Elements shall be encoded according to ITU-T Rec. Q.931, except where modified in this Recommendation. However, ITU-T Rec. Q.931 shall always dictate the proper ordering of information elements within a message, regardless of the order of elements listed within this Recommendation.

Intermediate systems (gateways and gatekeepers) shall follow the rules below with regard to H.225.0 call signalling optional messages and information elements:

1) The gateway should, and the gatekeeper shall, after appropriate modification, forward all information elements (optional or mandatory) associated with mandatory H.225.0 call signalling messages either from the terminal to the gateway/terminal or in the reverse direction. This includes such information elements as User-user information and the Display information.

2) A gateway should forward all H.225.0 call signalling messages, including those containing H.450 APDUs and information elements, in both directions.

3) A gatekeeper shall forward all H.225.0 call signalling messages, including those containing H.450 APDUs and information elements, in both directions after appropriate modification. Note that the gatekeeper may act as a signalling element that can provide features (such as supplementary service features) and may therefore modify, terminate, or originate H.225.0 call signalling messages.

H.323 gateways may be capable of converting H.450-series supplementary services and H.225.0 messages to the corresponding supplementary services and messages of ISO/IEC 11582, ISUP and other SCN signalling standards. Details are the scope of ITU-T Rec. H.246 and its annexes.

H.323 gateways may be capable of passing on unmodified signalling messages of ISO/IEC 11582, ISUP and other SCN signalling standards using tunnelling of non-H.323 signalling in H.225.0. Details are in Annex M/H.323 (see M.1/H.323, M.2/H.323, etc.).

In this version of this Recommendation, all references are to the 1998 version of ITU-T Rec. Q.931. The procedures of 3.1/Q.931 for circuit mode connection setup are followed. However, the implementor is reminded that although "bearer" is being signalled for, no actual "B-channels" of the ISDN type exist on the packet-based network side. Successful completion of the "call" results in an end-to-end reliable channel supporting H.245 messaging. Actually "bearer" setup is done using H.245. However, the use of Q.931 on the packet-based network side enables interworking with Q.931 on the SCN side, as well as providing a well-tested framework for general connection oriented calling features.

In general, the symmetric procedures of Annex D/Q.931 are used. This implies that the Q.931 state machine is followed as per Annex D/Q.931 with the exception that the procedure of D.3/Q.931 (Call collisions) shall not be followed; recovery from this glare condition is left to the application layer.

Endpoints not supporting Q.931 shifted code sets shall ignore all Q.931 messages using such methods.

Table 4 shows what messages are mandatory and optional for H.323 and H.225.0 call setup using Q.931 on the packet-based network.

Table 4/H.225.0 – H.225.0 usage of Q.931/Q.932 messages

	Transmit **(M, F, O, CM) (Note 1)**	**Receive and act on** **(M, F, O (Note 2), CM)**
Call establishment messages		
Alerting	M	M
Call Proceeding	O	CM (Notes 3 and 6)
Connect	M	M
Connect Acknowledge	F	F
Progress	O	CM (Note 6)
Setup	M	M
Setup Acknowledge	O	O
Call Clearing messages		
Disconnect	F	F
Release	F	F
Release Complete	M (Note 4)	M
Call Information Phase messages		
Resume	F	F
Resume Acknowledge	F	F
Resume Reject	F	F
Suspend	F	F
Suspend Acknowledge	F	F
Suspend Reject	F	F
User Information	O	O
Miscellaneous messages		
Congestion Control	F	F
Information	O	CM (Note 6)
Notify	O	O
Status	M (Note 5)	M
Status Inquiry	O	M

Table 4/H.225.0 – H.225.0 usage of Q.931/Q.932 messages

	Transmit **(M, F, O, CM) (Note 1)**	**Receive and act on** **(M, F, O (Note 2), CM)**
Q.932/H.450 messages		
Facility	M	M
Hold	F	F
Hold Acknowledge	F	F
Hold Reject	F	F
Retrieve	F	F
Retrieve Acknowledge	F	F
Retrieve Reject	F	F

NOTE 1 – M: Mandatory, F: Forbidden, O: Optional, CM: Conditionally Mandatory. Something is CM if it is required once an option is supported.

NOTE 2 – Note that Status shall not be sent in response to a message listed here as "O"; the receiver shall simply ignore the message if it does not support it.

NOTE 3 – Terminals intended to use gateways shall receive and act on Call Proceeding.

NOTE 4 – Release Complete is required to close the H.225.0 reliable call signalling channel. However, the call signalling channel shall remain open if other calls using the same call signalling channel are still in progress. Additionally, the Gatekeeper may set the **maintainConnection** flag to TRUE to prevent the closure of the call signalling channel.

NOTE 5 – The endpoint shall respond to an unknown message with a Status message; response to Status Inquiry is also mandatory. However, an endpoint is not required to send Status Inquiry. As a practical matter, the endpoint should be able to understand a Status message received in response to a message sent that was not known to the receiver.

NOTE 6 – Endpoints that support optional features that use these messages (such as H.245 tunnelling, H.450 supplementary services, tunnelling of signalling protocols, or features that use **genericData**) shall process these messages.

7.2 Common Q.931 information elements

7.2.1 Header information elements

For all H.225.0 call signalling messages, there are three common fields that are mandatory in addition to the message type that are described in this clause.

7.2.1.1 Protocol discriminator

As defined in 4.2/Q.931.

Shall be set to 08H – this identifies the message as Q.931/I.451 user-network message (encoded following Figure 4-2/Q.931). If a gatekeeper is acting as a network to supply supplementary services, it may be appropriate to use another value. This is for further study.

7.2.1.2 Call reference

As defined in 4.3/Q.931.

A call reference value length of two octets shall be supported by any H.323 endpoint.

The call reference value is chosen at the side originating the call and has to be locally unique. For subsequent communication, the calling and the called side shall use this call reference value in all the messages belonging to this particular call.

The value is encoded following Figure 4-5/Q.931 for a two-octet call reference value. The most significant octet of the reference value is always encoded in octet No. 2.

Note that the CRV is only unique on a particular part of a call, e.g., between two terminals, or between a terminal and a gatekeeper. If a given terminal has two calls in the same conference, each shall have the same conference ID but different CRVs.

The call reference flag shall be set according to the procedures described in ITU-T Rec. Q.931.

Note that the CRV values passed in RAS messages shall conform to the structure as specified in ITU-T Rec. Q.931. Specifically, the call reference flag shall be included as the most significant bit of the Call ReferenceValue. This restricts the actual CRV to the range of 0 through 32 767, inclusive.

The Global Call Reference, as shown Figure 4-5/Q.931 and having the numeric value 0, is used to refer to all calls on the Call Signalling Channel or the RAS channel.

7.2.1.3 Message type

The message type is encoded according to Figure 4-6/Q.931 using the values specified in Table 4-2/Q.931. H.225.0 specific extensions are for further study.

7.2.2 Message-specific information elements

The general encoding rules for the following information elements are defined in 4.5.1/Q.931 and Table 4-3/Q.931. These rules shall be followed. The escape mechanism (see Figure 4-8/Q.931) is optional.

7.2.2.1 Bearer capability

This information element is encoded according to Figure 4-11/Q.931 and Table 4-6/Q.931. If this information element is received in a packet-based network-to-packet-based network call, it may be ignored by the receiver. If this information element appears in a Setup message for a call-independent signalling connection as defined in ITU-T Rec. H.450.1, the coding shall follow 7.2.2.1.2. In all other cases, coding shall follow 7.2.2.1.1. The octet number references refer to Figure 4-11/Q.931.

7.2.2.1.1 Bearer capability default encoding

H.323 entities shall encode the Bearer capability IE as follows, unless indicated otherwise in subsequent clauses.

Extension bit for octet No. 3 (bit 8)

– Shall be set to "1".

Coding Standard (octet No. 3, bits 6-7)

– Shall be set to "00" indicating "ITU-T".

Information transfer capability (octet No. 3, bits 1-5)

– For calls originating from an ISDN endpoint, the information indicated to the gateway shall be forwarded.

NOTE – This is to allow some advance information about the nature of the connection to be forwarded to the H.323 endpoint, e.g., voice only vs. data vs. video; this would have an impact on the bandwidth required as well as on the ability/willingness to accept the call or not.

– Calls that originate from an H.323 endpoint shall use this field to indicate their wish to place an audiovisual call. Therefore, the field shall be set either to "unrestricted digital information", i.e., "01000" or to "restricted digital information", i.e., "01001". If a speech only call is to be placed, the H.323 terminal shall set the information transfer capability to either "speech" (i.e., "00000") or to "3.1 kHz audio" (i.e., "10000").

Extension bit for octet No. 4 (bit 8)

– Shall be set to "0" if the information transfer rate is set to "multirate"; shall be set to "1" otherwise.

Transfer Mode (octet No. 4, bits 6, 7)

– Shall specify "circuit mode", value "00".

Information transfer rate (octet No. 4, bits 1-5)

– Shall be encoded following Table 4-6/Q.931 except that the value "00000" (for packet mode) is not permitted unless the gateway is connected to a packet network.

Rate multiplier (octet No. 4.1)

– Shall be present if information transfer rate is set to "multirate".

– The extension bit (bit 8) shall be set to "1".

– The bits 1 through 7 shall indicate the bandwidth needed for the call as defined in the following (note, that in contrast to ITU-T Rec. Q.931, a value of "0000001" is allowed here).

– For a call originating from an ISDN endpoint, the gateway shall simply pass on the information that it receives from the ISDN.

– For a call incoming from an H.324 endpoint, the gateway shall set the rate multiplier to 01H.

– For a call incoming from B-ISDN, some translation from ITU-T Rec. Q.2931 to ITU-T Rec. Q.931 needs to be performed. This is for further study.

– For a call originated from an H.323 endpoint, this shall be used to indicate the bandwidth to be used for this call. If the called system is another H.323 endpoint, this value may reflect the bandwidth to be used on the packet-based network, but the receiving terminal is not required to follow this information. If a gateway is involved, then this value shall reflect the number of external connections to be set up. The bandwidth needed for the call is the bandwidth needed on the SCN side, and may or may not match the bandwidth allowed on the packet-based network by the ACF/BCF messages.

Layer 1 protocol (Octet No. 5)

– The extension bit (bit 8) shall be set to "1".

– Bits 6 and 7 shall indicate the layer one identifier, i.e., "01".

– Bits 1 through 5 shall indicate the layer one protocol.

– The allowed values are G.711 (A-law "00011" and μ-law "00010") to indicate a voice-only call and H.221 and H.242 ("00101") to indicate an H.323 videophone call.

Octets Nos. 5a, 5b, 5c, 5d, 6 and 7 shall not be present.

7.2.2.1.2 Bearer capability encoding for H.450.1 call-independent signalling connections

H.323 entities shall encode the Bearer capability IE as follows for call-independent signalling connections, as defined in ITU-T Rec. H.450.1.

Extension bit for octet No. 3 (bit 8)

– Shall be set to "1".

Coding standard (octet No. 3, bits 6-7)

– Shall be set to "01" indicating "Other international standard". Note that when this coding standard is indicated, the coding defined in ITU-T Rec. Q.931 shall apply for octets 1 to 2 and bit 8 of octets 3 to 4. Information transfer capability, Transfer mode and Information transfer rate shall be encoded as indicated and no other octets shall be included.

Information transfer capability (octet No. 3, bits 1-5)

– Shall be set to "01000", indicating "Unrestricted digital information".

Extension bit for octet No. 4 (bit 8)

– Shall be set to "1".

Transfer mode (octet No. 4, bits 6, 7)

– Shall be set to "00", indicating "Call-independent signalling connection".

Information transfer rate (octet No. 4, bits 1-5)

– Shall be set to "00000", indicating "Call-independent signalling connection".

Octets 4.1 and higher shall not be included.

7.2.2.2 Call identity

The possible use of the Call identity IE is for further study. This study should consider multi-stage dialling, including terminal-to-gatekeeper-to-terminal and terminal-to-gateway-to-terminal, and loose source routing.

7.2.2.3 Call state

This information element is encoded following Figure 4-13/Q.931.

Octet No. 3 coding standard (bits 8-7)

– Set to "00" to indicate ITU-T standardized coding.

Call state value (octet No. 3, bits 1-6)

– Set as per Table 4-8/Q.931 but do not use the global interface state values. Values are interpreted as User State as per use of Annex D/Q.931. Note that most of the listed codes will not be generated by an H.323 terminal.

7.2.2.4 Called party number

This information element is encoded following Figure 4-14/Q.931 and Table 4-9/Q.931.

Octet No. 3 extension (bit 8)

– Set to "1".

Type of number (octet No. 3, bits 5-7)

– Encoded following the values and rules of Table 4-9/Q.931.

Numbering plan identification (octet No. 3, bits 1-4)

– Encoded following the values and rules of Table 4-9/Q.931. A number in the form of a dialled digit string should be coded as "0000" (Unknown). If set to "1001" (Private Numbering Plan) in a packet-based network originated call, this indicates that:

  1) the dialled digit string is not present in Setup; and

  2) the call will be routed via an alias address in the User-user information.

Type of number (octet No. 3, bits 5-7)

– Encoded following the values and rules of Table 4-9/Q.931. A number with the Numbering plan identification coded as "0000" (Unknown) shall be coded as "000" (Unknown). A number with the Numbering plan identification coded as "0001" (ISDN/Telephony Numbering Plan, ITU-T Rec. E.164) with the Type of number coded as "000" (Unknown) may be used for backward compatibility.

Number "digits"

– Any number of IA5 characters, according to the formats specified in the appropriate numbering/dialling plan.

NOTE – An E.164 number shall only consist of IA5 characters "0", "1", "2", "3", "4", "5", "6", "7", "8", "9" and "0".

7.2.2.5 Called party subaddress

Use as per ITU-T Rec. Q.931.

7.2.2.6 Calling party number

This information element is encoded following Figure 4-16/Q.931 and Table 4-11/Q.931.

Type of number (octet No. 3, bits 5-7)

– Encoded following the values and rules of Table 4-11/Q.931. A number with the Numbering plan identification coded as "0000" (Unknown) shall be coded as "000" (Unknown). A number with the Numbering plan identification coded as "0001" (ISDN/Telephony Numbering Plan, ITU-T Rec. E.164) with the Type of number coded as "000" (Unknown) may be used for backward compatibility.

Numbering plan identification (octet No. 3, bits 1-4)

– Encoded following the values and rules of Table 4-11/Q.931. A number in the form of a dialled digit string should be coded as "0000" Unknown. If set to "1001" (Private Numbering Plan) in a packet-based network originated call, this indicates that:

1) the dialled digit string is not present in Setup; and
2) the call will be routed via an alias address in the User-user information.

Octet No. 3a

– Encoded following the values and rules of Table 4-11/Q.931.

Number "digits"

– Any number of IA5 characters, according to the formats specified in the appropriate numbering/dialling plan.

NOTE – An E.164 number shall only consist of IA5 characters "0", "1", "2", "3", "4", "5", "6", "7", "8", "9" and "0".

H.323 endpoints shall not send multiple Calling Party Number IEs in the same message. Gateways may provide support for interworking with Q.931 SETUP messages that contain multiple Calling Party Number IEs. Gateways that provide such support shall map the first Q.931 Calling Party Number IE to the Calling Party Number IE of the H.225.0 Setup message, and map subsequent Q.931 Calling Party Number IEs to the **additionalSourceAddresses** field of the H.225.0 Setup message.

7.2.2.7 Calling party subaddress

Use as per ITU-T Rec. Q.931.

7.2.2.8 Cause

If received, the rules defined in ITU-T Rec. Q.850 apply. Note that either Cause or **ReleaseCompleteReason** is mandatory for Release Complete; the Cause IE is optional elsewhere. The Cause IE and the **ReleaseCompleteReason** (a part of the Release Complete message) are mutually exclusive. Gateways shall map from a **ReleaseCompleteReason** to the Cause IE when sending a Release Complete message to the circuit-switched side from the packet-based network side (see Table 5). (The reverse mapping is not required as packet-based network entities are required to decode the Cause IE.)

Table 5/H.225.0 – ReleaseCompleteReason to Cause IE mapping

ReleaseCompleteReason code	Corresponding Q.931/Q.850 cause value
noBandwidth	34 – No circuit/channel available
gatekeeperResources	47 – Resource unavailable, unspecified
unreachableDestination	3 – No route to destination
destinationRejection	16 – Normal call clearing
invalidRevision	88 – Incompatible destination
noPermission	127 – Interworking, unspecified
unreachableGatekeeper	38 – Network out of order
gatewayResources	42 – Switching equipment congestion
badFormatAddress	28 – Invalid number format (address incomplete)
adaptiveBusy	41 – Temporary Failure
inConf	17 – User busy
undefinedReason	31 – Normal, unspecified
facilityCallDeflection	16 – Normal call clearing
securityDenied	31 – Normal, unspecified
securityWrongSyncTime	31 – Normal, unspecified
securityReplay	31 – Normal, unspecified
securityWrongGeneralID	31 – Normal, unspecified
securityWrongSendersID	31 – Normal, unspecified
securityMessageIntegrityFailed	31 – Normal, unspecified
securityWrongOID	31 – Normal, unspecified
securityDHmismatch	31 – Normal, unspecified
securityCertificateExpired	31 – Normal, unspecified
securityCertificateDateInvalid	31 – Normal, unspecified
securityCertificateRevoked	31 – Normal, unspecified
securityCertificateNotReadable	31 – Normal, unspecified
securityCertificateSignatureInvalid	31 – Normal, unspecified
securityCertificateMissing	31 – Normal, unspecified
securityCertificateIncomplete	31 – Normal, unspecified
securityUnsupportedCertificateAlgOID	31 – Normal, unspecified
securityUnknownCA	31 – Normal, unspecified
calledPartyNotRegistered	20 – Subscriber absent
callerNotRegistered	31 – Normal, unspecified
newConnectionNeeded	47 – Resource unavailable, unspecified
nonStandardReason	127 – Interworking, unspecified
replaceWithConferenceInvite	31 – Normal, unspecified
genericDataReason	31 – Normal, unspecified
neededFeatureNotSupported	31 – Normal, unspecified
tunnelledSignallingRejected	127 – Interworking, unspecified
InvalidCID	3 – No route to destination
hopCountExceeded	3 – No route to destination

Gateways shall also map from **AdmissionRejectReason** and **LocationRejectReason** to the Cause IE when sending a Release Complete message to the circuit-switched side after receiving an **AdmissionReject** or a **LocationReject** (Table 6).

Table 6/H.225.0 – AdmissionRejectReason/LocationRejectReason to Cause IE mapping

AdmissionRejectReason or LocationRejectReason code	Corresponding Q.931/Q.850 cause value
calledPartyNotRegistered	20 – Subscriber absent
invalidPermission	127 – Interworking, unspecified
requestDenied	31 – Normal, unspecified
undefinedReason	31 – Normal, unspecified
callerNotRegistered	31 – Normal, unspecified
routeCallToGatekeeper	Not applicable
invalidEndpointIdentifier	127 – Interworking, unspecified
resourceUnavailable	47 – Resource unavailable, unspecified
securityDenial	31 – Normal, unspecified
qosControlNotSupported	63 – Service or option not available, unspecified
incompleteAddress	28 – Invalid number format
aliasesInconsistent	31 – Normal, unspecified
routeCallToSCN	3 – No route to destination
exceedsCallCapacity	41 – Temporary Failure
collectDestination	31 – Normal, unspecified
collectPIN	31 – Normal, unspecified
genericDataReason	31 – Normal, unspecified
neededFeatureNotSupported	31 – Normal, unspecified
securityWrongSyncTime	31 – Normal, unspecified
securityReplay	31 – Normal, unspecified
securityWrongGeneralID	31 – Normal, unspecified
securityWrongSendersID	31 – Normal, unspecified
securityIntegrityFailed	31 – Normal, unspecified
securityWrongOID	31 – Normal, unspecified
secuirtyDHMismatch	31 – Normal, unspecified
noRouteToDestination	3 – No route to destination
unallocatedNumber	1 – Unallocated number

7.2.2.9 Channel identification

Use is for further study; may be used to provide feedback on multiple call attempts.

7.2.2.10 Connected number

Encoded following 4.1/Q.951.5.

7.2.2.11 Connected sub-address

Encoded following 4.2/Q.951.5.

7.2.2.12 Congestion level

Shall not be used.

7.2.2.13 Date/time

Encoded following Figure 4-21/Q.931.

7.2.2.14 Display

Encoded following Figure 4-22/Q.931. The maximum length of the entire information element is 82 octets.

7.2.2.15 Extended facility information element

Any Extended Facility IE that is used to indicate unmodified semantics as defined in Q.95.x-series Recommendations shall be encoded following 8.2.4/Q.932. In this case, the Service ADUs shall be formed according to ROSE (uses ITU-T Rec. X.680 (Specification of ASN.1) and ITU-T Rec. X.690 (Specification of basic encoding rules for ASN.1)) as defined in ITU-T Rec. X.229.

7.2.2.16 Facility

In order to signal call redirection specific to H.323 procedures (call forwarding, redirecting a call to the MC, or forcing a call to be routed to the gatekeeper) or in case of supplementary service signalling according to ITU-T Rec. H.450, the User-user information element of the Facility message is used. This particular case shall be indicated by coding a Facility IE of length zero; i.e., the Facility information element shall consist of exactly 2 octets as follows:

- Octet No. 1 (information element identifier) shall be set to "00011100" ("1C'H) to indicate the Facility IE.
- Octet No. 2 (information element length) shall be set to "0" to indicate that no further octets belonging to this information element follow.

In order to indicate call forwarding, the Facility IE shall be empty and the **Facility-UUIE** shall indicate in the **alternativeAddress** or the **alternativeAliasAddress** the terminal to which the call is to be redirected. In this case, the **facilityReason** shall be set to **callForwarded**.

To instruct an endpoint to call a different endpoint because the calling endpoint wishes to join a conference and the called endpoint does not have the MC, the Facility IE would be left empty as well. The **conferenceID** shall indicate the conference to join and the reason in the **Facility-UUIE** shall be **routeCallToMC**.

Also, to instruct the calling endpoint to signal the called endpoint through the called endpoint's gatekeeper, the Facility IE is left empty. The **conferenceID** in the **Facility-UUIE** shall indicate the conference to join and the reason in the **Facility-UUIE** shall be **routeCallToGatekeeper**.

Any Facility IE that is used to indicate unmodified semantics as defined in Q.95.x-series Recommendations shall be encoded following 8.2.3/Q.932. In this case, the Service ADUs shall be formed according to ROSE (uses ITU-T Rec. X.680 (Specification of ASN.1) and ITU-T Rec. X.690 (Specification of basic encoding rules for ASN.1)) as defined in ITU-T Rec. X.229.

7.2.2.17 High layer compatibility

FFS.

7.2.2.18 Keypad facility

Encoded following Figure 4-24/Q.931. The use of the exclamation point character "!" shall represent a hookflash indication. Endpoints not supporting reception of the hookflash indication shall ignore the "!" if received.

7.2.2.19 Low layer compatibility

FFS.

7.2.2.20 More data

Shall not be used.

7.2.2.21 Network-specific facilities

Shall not be used.

7.2.2.22 Notification indicator

Encoded following 4.5.22/Q.931.

7.2.2.23 Progress indicator

Encoded following Figure 4-29/Q.931 and Table 4-20/Q.931.

This information element is only required for interfacing an H.323 terminal to an ISDN- and ATM-based terminal where detailed call proceeding information is available. In this case, the gateway shall forward this information to the H.323 terminal. The H.323 end system need not interpret this information element.

If this information element is generated by an H.323 terminal, the following restrictions apply:

Coding standard (octet No. 3, bit 6, 7)

– Shall indicate "ITU-T" ("00").

Location

– Following Table 4-20/Q.931.

– The values "user" ("0000"), "private network serving the local user" ("0001"), and "private network serving the remote user" ("0101") are permitted.

Progress description

– Following Table 4-20/Q.931.

7.2.2.24 Redirecting number

Encoded following 4.6.7/Q.931. Note that this IE is provided only to facilitate interworking with the SCN, and not to provide a mechanism for H.323-based call diversion services. Call diversion services in H.323 are defined by ITU-T Rec. H.450.3.

7.2.2.25 Repeat indicator

Shall not be used.

7.2.2.26 Restart indicator

Shall not be used.

7.2.2.27 Segmented message

Shall not be used. Note that there is no critical upper limit on the message size in ITU-T Rec. H.323 and this Recommendation.

7.2.2.28 Sending complete

Encoded following Figure 4-33/Q.931.

No restrictions apply.

7.2.2.29 Signal

Encoded following Figure 4-34/Q.931 and Table 4-24/Q.931.

No restrictions apply.

7.2.2.30 Transit network selection

Shall not be used.

7.2.2.31 User-user

Encoded following Figure 4-36/Q.931 and Table 4-26/Q.931, as modified here.

The User-user information element shall be used by all H.323 entities to convey H.323-related information. Actual user-user information to be exchanged only between the involved terminals is nested in the **user-data** field of the **H323-UserInformation** PDU (to which no restrictions apply).

The following restrictions apply:

Length of user-user contents

- Shall be 2 octets instead of 1 (as in Figure 4-36/Q.931).

Protocol discriminator

- Shall indicate ITU-T Recs X.680 and X.690 (ASN.1) coded user information ("00000101").

  NOTE – This is taken from the 1998 revision of ITU-T Rec. Q.931 that references the earlier revisions of ASN.1. The correct references to ASN.1 are ITU-T Recs X.680 (syntax) and X.691 (PER).

User information

- Shall contain an ASN.1 structure (**H323-UserInformation**) that, besides the H.323 relevant information, includes the actual user data as follows. The ASN.1 is encoded using the aligned variant of the packed encoding rules as specified in ITU-T Rec. X.691.

The **H323-UserInformation** structure contains the **h323-uu-pdu** and **user-data** fields.

The **h323-uu-pdu** field of the **H323-UserInformation** structure contains the following fields. Note that not all fields in **h323-uu-pdu** are permitted in every message. See the description of each individual message for restrictions.

- **h323-message-body** – This field contains information specific to a particular H.225.0 signalling message, as described in 7.3 and 7.4. A sender may select a choice of **empty** if there is no need to send the UUIE field (**Facility-UUIE**, etc.) in a particular message, such as when a Facility message is used to transport non-call associated information. Note that beginning with version 4 of this Recommendation, if a message is associated with a particular call, then the sender shall include the UUIE field. This is necessary in order to provide the **callIdentifier** field.
- **nonStandardData** – This field carries information not defined in this Recommendation (for example, proprietary data).
- **h4501SupplementaryService** – This field carries a sequence of H4501SupplementaryService APDUs as defined in Table 3/H.450.1.
- **h245Tunnelling** – This element is set to TRUE if tunnelling of H.245 messages is enabled. Systems compliant with H.225.0 version 4 or higher shall set this element to TRUE if the Fast Connect procedure is used to establish the call.
- **h245Control** – This field carries a sequence of tunnelled H.245 PDUs. Each octet string shall contain exactly one H.245 PDU.
- **nonStandardControl** – This field contains control information not defined in this Recommendation (for example, proprietary control information).

- **callLinkage** – The contents of this field are typically controlled by a call linkage service. For the procedures and semantics of this field refer to ITU-T Rec. H.323.
- **tunnelledSignallingMessage** – A tunnelled entire signalling message in its native format to support additional end-to-end call control signalling. The **tunnelledProtocolID** field identifies the protocol being tunnelled. The **messageContent** field is a sequence of actual entire tunnelled messages in their native binary format; this allows aggregation of tunnelled messages in one H.225.0 messsage. If the **tunnellingRequired** field is present, the call shall only proceed if tunnelling is supported.
- **provisionalRespToH245Tunnelling** – This flag is used to signal that the called entity has not yet decided whether H.245 tunnelling is applicable for this call. If present, the **h245Tunnelling** flag shall be ignored by the receiving entity.
- **stimulusControl** – This field is reserved for future use by the ITU-T for a stimulus-based protocol.
- **genericData** – This field is a list of generic elements related to features that are defined outside of the base H.225.0 specification. These parameters may be used, for example, for tunnelling information transparently through H.225.0.

The **user-data** field of the **H323-UserInformation** structure contains the following fields:

- **protocol-discriminator** – This field is encoded following Table 4-26/Q.931.
- **user-information** – This field is encoded following 4.5.30/Q.931.

7.3 Q.931-based H.225.0 call signalling message details

Note that the lengths of the information elements specified in the tables below refer to messages that are generated by H.323 terminals only. The size of the User-user information element shown is understood as the size of the **user-data** structure in **H323-UserInformation** and does not include the **h323-UU-PDU**. The total size of **H323-UserInformation** is limited to 65 536 octets. Regardless of the specified sizes, messages forwarded from the SCN side may have different (larger) sizes.

Also note that an information element specified below as mandatory, optional, or forbidden, refers only to whether or not H.323 terminals may originate such information elements.

7.3.1 Alerting

This message may be sent by the called user to indicate that called user alerting has been initiated. In everyday terms, the "phone is ringing."

Follow Table 3-2/Q.931 (1998 version) as modified below in Table 7.

Table 7/H.225.0 – Alerting

Information element	H.225.0 status (M/F/O)	Length in H.225.0
Protocol discriminator	M	1
Call reference	M	3
Message type	M	1
Bearer capability	O	5-6
Extended facility	O	8-*
Channel identification	FFS	NA
Facility	O	8-*
Progress indicator	O	2-4
Notification indicator	O	2-*

Table 7/H.225.0 – Alerting

Information element	H.225.0 status (M/F/O)	Length in H.225.0
Display	O	2-82
Signal	O	2-3
High layer compatibility	FFS	NA
User-user	M	2-131

The User-user information element contains the Alerting-UUIE defined in the H.225.0 Message Syntax. The **Alerting-UUIE** includes the following:

protocolIdentifier – Set to the version of H.225.0 supported.

destinationInfo – Contains an **EndpointType** to allow the caller to determine whether the call involves a gateway or not.

h245Address – This is a specific transport address on which the called endpoint or gatekeeper handling the call would like to establish H.245 signalling. This address may also be sent in Call Proceeding, Progress, or Connect.

callIdentifier – A globally unique call identifier set by the originating endpoint which can be used to associate RAS signalling with the modified Q.931 signalling used in this Recommendation.

h245SecurityMode – An H.323 entity that receives a Setup message with the **h245SecurityCapability** set shall respond with the corresponding, acceptable **h245SecurityMode** in the Call Proceeding, Alerting, Progress, or Connect**.**

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

fastStart – Used only in the fast connect procedure, **fastStart** supports the signalling needed to open a logical channel. This uses the **OpenLogicalChannel** structure defined in ITU-T Rec. H.245, but the sender of this indicates the modes it prefers to receive and transmit, and the transport addresses where it expects to receive media streams.

multipleCalls – If TRUE, this indicates that the sender of the message is capable of signalling multiple calls over a single call signalling connection.

maintainConnection – If TRUE, this indicates that the sender of the message is capable of supporting a signalling connection when no calls are currently signalled over the connection.

alertingAddress – Contains the alias addresses for the alerting party.

presentationIndicator – Indicates whether presentation of the **alertingAddress** should be allowed or restricted.

screeningIndicator – Indicates whether the **alertingAddress** was provided by the endpoint or network (gatekeeper), and whether the **alertingAddress** was screened by a gatekeeper.

fastConnectRefused – A called endpoint should return this element in any message up to and including the Connect message when establishing a call to indicate that it refuses the Fast Connect procedure.

serviceControl – Contains service-specific data, or references to it, that may be used as part of the setup procedure by the calling endpoint (e.g., a menu of options for call diversion) as described, for example, in Annex K/H.323.

capacity – This field indicates the sending endpoint's available call capacity at this point in time, assuming that this Alerting message represents an active call. When sending this field, the endpoint shall include the **currentCallCapacity** element.

featureSet – This field specifies a set of generic features that relate to this call.

7.3.2 Call proceeding

This message may be sent by the called user to indicate that requested call establishment has been initiated and no more call establishment information will be accepted. See Table 8.

Table 8/H.225.0 – Call proceeding

Information element	H.225.0 status (M/F/O)	Length in H.225.0
Protocol discriminator	M	1
Call reference	M	3
Message type	M	1
Bearer capability	O	5-6
Extended facility	O	8-*
Channel identification	FFS	NA
Facility	O	8-*
Progress indicator	O	2-4
Notification indicator	O	2-*
Display	O	2-82
High layer compatibility	FFS	NA
User-user	M	2-131

The User-user information element contains the **CallProceeding-UUIE** defined in the H.225.0 Message Syntax. The **CallProceeding-UUIE** includes the following:

protocolIdentifier – Set to the version of H.225.0 supported.

destinationInfo – Contains an **EndpointType** to allow the caller to determine whether the call involves a gateway or not.

h245Address – This is a specific transport address on which the called endpoint or gatekeeper handling the call would like to establish H.245 signalling.

callIdentifier – A globally unique call identifier set by the originating endpoint which can be used to associate RAS signalling with the modified Q.931 signalling used in this Recommendation.

h245SecurityMode – An H.323 entity that receives a Setup message with the **h245SecurityCapability** set shall respond with the corresponding, acceptable **h245SecurityMode** in the Call Proceeding, Alerting, Progress, or Connect.

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

fastStart – Used only in the fast connect procedure, **fastStart** supports the signalling needed to open a logical channel. This uses the **OpenLogicalChannel** structure defined in ITU-T Rec. H.245, but the sender of this indicates the modes it prefers to receive and transmit, and the transport addresses where it expects to receive media streams.

multipleCalls – If TRUE, this indicates that the sender of the message is capable of signalling multiple calls over a single call signalling connection.

maintainConnection – If TRUE, this indicates that the sender of the message is capable of supporting a signalling connection when no calls are currently signalled over the connection.

fastConnectRefused – A called endpoint should return this element in any message up to and including the Connect message when establishing a call to indicate that it refuses the Fast Connect procedure.

featureSet – This field specifies a set of generic features that relate to this call.

7.3.3 Connect

This message shall be sent by the called entity to the calling entity (gatekeeper, gateway, or calling terminal) to indicate acceptance of the call by the called entity. Follow Table 3-4/Q.931, as modified in Table 9 below.

Table 9/H.225.0 – Connect

Information element	H.225.0 status (M/F/O)	Length in H.225.0
Protocol discriminator	M	1
Call reference	M	3
Message type	M	1
Bearer capability	O	5-6
Extended facility	O	8-*
Channel identification	FFS	NA
Facility	O	8-*
Progress indicator	O	2-4
Notification indicator	O	2-*
Display	O	2-82
Date/Time	O	8
Connected Number	O	2-*
Connected Subaddress	O	2-23
Low layer compatibility	FFS	NA
High layer compatibility	FFS	NA
User-user	M	2-131

The User-user information element contains the **Connect-UUIE** defined in the H.225.0 Message Syntax. The **Connect-UUIE** includes the following:

protocolIdentifier – Set by the called endpoint to the version of H.225.0 supported.

h245Address – This is a specific transport address on which the called endpoint or gatekeeper handling the call would like to establish H.245 signalling. This address shall be sent if sent earlier in Alerting, Progress, or Call Proceeding.

destinationInfo – Contains an **EndpointType** to allow the caller to determine whether the call involves a gateway or not.

conferenceID – Will contain a unique number to allow the conference to be uniquely identified from all others as received in the Setup.

callIdentifier – A globally unique call identifier set by the originating endpoint which can be used to associate RAS signalling with the modified Q.931 signalling used in this Recommendation.

h245SecurityMode – An H.323 entity that receives a Setup message with the **h245SecurityCapability** set shall respond with the corresponding, acceptable **h245SecurityMode** in the Call Proceeding, Alerting, Progress, or Connect.

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

fastStart – Used only in the fast connect procedure, **fastStart** supports the signalling needed to open a logical channel. This uses the **OpenLogicalChannel** structure defined in ITU-T Rec. H.245, but the sender of this indicates the modes it prefers to receive and transmit, and the transport addresses where it expects to receive media streams.

multipleCalls – If TRUE, this indicates that the sender of the message is capable of signalling multiple calls over a single call signalling connection.

maintainConnection – If TRUE, this indicates that the sender of the message is capable of supporting a signalling connection when no calls are currently signalled over the connection.

language – Indicates the language(s) in which the user would prefer to receive announcements and prompts. The field contains one or more RFC 1766 compliant language tags.

connectedAddress – Contains the alias addresses for the connected (answering) party; the dialled digit string of the connected party is in the Connected Number IE.

presentationIndicator – Indicates whether presentation of the **connectedAddress** should be allowed or restricted. If both **presentationIndicator** and the presentation indicator of the Connected Number IE are present and are in conflict, the presentation indicator of the Connected Number IE shall be used.

screeningIndicator – Indicates whether the **connectedAddress** was provided by the endpoint or network (gatekeeper), and whether the **connectedAddress** was screened by a gatekeeper. If both **screeningIndicator** and the screening indicator of the Connected Number IE are present and are in conflict, the screening indicator of the Connected Number IE shall be used.

fastConnectRefused – A called endpoint should return this element in any message up to and including the Connect message when establishing a call to indicate that it refuses the Fast Connect procedure.

serviceControl – Contains service-specific data, or references to it, that could be used by an endpoint or gateway (e.g., for displaying a menu of options to a caller) as described, for example, in Annex K/H.323.

capacity – This field indicates the sending endpoint's available call capacity at this point in time, assuming that this Connect message represents an active call. When sending this field, the endpoint shall include the **currentCallCapacity** element.

featureSet – This field specifies a set of generic features that relate to this call.

7.3.4 Connect acknowledge

This message shall not be sent.

7.3.5 Disconnect

This message shall not be sent by an H.323 entity.

The contents and semantics of a Disconnect message received from the network are defined in Table 3-6/Q.931 and in 10.5 of ISO/IEC 11582.

7.3.6 Information

This message may be sent to provide additional information. It may be used to provide information for call establishment (e.g., overlap sending) or miscellaneous call-related information. It may be used to deliver proprietary features.

This message may be sent by an H.323 entity.

This message follows Table 3-7/Q.931 with the modifications as shown in Table 10.

Table 10/H.225.0 – Information message content

Information element	H.225.0 status (M/F/O)	Length in H.225.0
Protocol discriminator	M	1
Call reference	M	3
Message type	M	1
Sending complete	O	1
Display	O	2-82
Keypad facility	O	2-34
Signal	O	2-3
Called party number	O (Note)	2-35
User-user	M	2-131
NOTE – The Called Party Number IE will be used to carry numbers from a Private Numbering Plan when performing overlapped sending according to 8.1.12/H.323.		

The User-user information element contains the **Information-UUIE** defined in the H.225.0 Message Syntax. The **Information-UUIE** includes the following:

protocolIdentifier – Set to the version of H.225.0 supported.

callIdentifier – A globally unique call identifier set by the originating endpoint which can be used to associate RAS signalling with the modified Q.931 signalling used in this Recommendation.

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

fastStart – This field shall not be included, and shall be ignored upon receipt.

fastConnectRefused – This field shall not be included, and shall be ignored upon receipt.

circuitInfo – This field provides information about the SCN circuit or circuits used for this call.

7.3.7 Progress

This message may be sent by an H.323 gateway to indicate the progress of a call in the event of interworking with SCN. This message may also be sent by an H.323 endpoint before the Connect message, depending on supplementary service interaction.

Follow Table 3-9/Q.931 and 10.10 of ISO/IEC 11582 as modified in Table 11.

Table 11/H.225.0 – Progress

Information element	H.225.0 status (M/F/O)	Length in H.225.0
Protocol discriminator	M	1
Call reference	M	3
Message type	M	1
Bearer capability	O	5-6
Cause	O	2-32
Extended facility	O	8-*
Channel identification	FFS	NA
Facility	O	8-*
Progress indicator	M	2-4
Notification indicator	O	2-*
Display	O	2-82
High layer compatibility	FFS	NA
User-user	M	2-131

The User-user information element contains the **Progress-UUIE** defined in the H.225.0 Message Syntax. The **Progress-UUIE** includes the following:

protocolIdentifier – Set to the version of H.225.0 supported.

destinationInfo – Contains an **EndpointType** to allow the caller to determine whether the call involves a gateway or not.

h245Address – This is a specific transport address on which the called endpoint or gatekeeper handling the call would like to establish H.245 signalling. This address shall be sent if sent earlier in Call Proceeding, Alerting, or Connect.

callIdentifier – A globally unique call identifier set by the originating endpoint which can be used to associate RAS signalling with the modified Q.931 signalling used in this Recommendation.

h245SecurityMode – An H.323 entity that receives a Setup message with the **h245SecurityCapability** set shall respond with the corresponding, acceptable **h245SecurityMode** in the Call Proceeding, Alerting, Progress, or Connect.

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

fastStart – Used only in the fast connect procedure, **fastStart** supports the signalling needed to open a logical channel. This uses the **OpenLogicalChannel** structure defined in ITU-T Rec. H.245, but the sender of this indicates the modes it prefers to receive and transmit, and the transport addresses where it expects to receive media streams.

multipleCalls – If TRUE, this indicates that the sender of the message is capable of signalling multiple calls over a single call signalling connection.

maintainConnection – If TRUE, this indicates that the sender of the message is capable of supporting a signalling connection when no calls are currently signalled over the connection.

fastConnectRefused – A called endpoint should return this element in any message up to and including the Connect message when establishing a call to indicate that it refuses the Fast Connect procedure.

7.3.8 Release

This message shall not be sent by an H.323 entity.

The contents and semantics of a Release message received are defined in Table 3-10/Q.931 and in 10.5 of ISO/IEC 11582.

7.3.9 Release complete

This message shall be sent by a terminal to indicate release of the call. Afterwards, the Call Reference Value (CRV) is available for reuse.

The disconnect/release/release complete sequence is not used since its primary purpose is to indicate the completion of release of circuit-switched resources. As this does not apply to the packet-based network environment, the single step method of sending only Release Complete is used.

Follow Table 3-11/Q.931. Table 12 modifications apply.

Table 12/H.225.0 – Release complete

Information element	H.225.0 status (M/F/O)	Length in H.225.0
Protocol discriminator	M	1
Call reference	M	3
Message type	M	1
Cause	CM (Note)	2-32
Facility	O	8-*
Notification indicator	O	2-*
Display	O	2-82
Signal	O	2-3
User-user	M	2-131
NOTE – Either the Cause IE or the **ReleaseCompleteReason** shall be present.		

If this message is sent in response to a Facility message with an empty Facility IE, the **ReleaseCompleteReason** shall be set to **facilityCallDeflection**.

If this message is forwarded from a SCN by a gateway, the cause value shall be set as specified in ITU-T Rec. Q.931.

The User-user information element contains the **ReleaseComplete-UUIE** defined in the H.225.0 Message Syntax. The **ReleaseComplete-UUIE** includes the following:

protocolIdentifier – Set to the version of H.225.0 supported.

reason – More information on why the call was released. A reason of **genericDataReason** indicates that the call was cleared as a result of a generic element or feature; in this case, additional information may be specified in the **genericData** field of the **h323-uu-pdu** of this message. A reason of **neededFeatureNotSupported** indicates that a feature required by one entity is not supported by another. A reason of **tunnelledSignallingRejected** is sent if the call is cleared because the sender does not allow tunnelled non-H.323 signalling, and tunnelling is required in order for the call to succeed. A reason of **hopCountExceeded** indicates that the call was rejected because the **hopCount** value reached 0 and hence the call cannot proceed further.

callIdentifier – A globally unique call identifier set by the originating endpoint which can be used to associate RAS signalling with the modified Q.931 signalling used in this Recommendation.

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

busyAddress – Contains the alias addresses for the busy party.

presentationIndicator – Indicates whether presentation of the **busyAddress** should be allowed or restricted.

screeningIndicator – Indicates whether the **busyAddress** was provided by the endpoint or network (gatekeeper), and whether the **busyAddress** was screened by a gatekeeper.

capacity – Indicates the sending endpoint's available call capacity after the call referenced in this Release Complete message has been released. When sending this field, the endpoint shall include the **currentCallCapacity** element.

serviceControl – Contains service-specific data, or references to it, for post-call services (e.g., an error message or announcement) as described, for example, in Annex K/H.323.

featureSet – This field specifies a set of generic features that relate to this call.

7.3.10 Setup

This message shall be sent by a calling H.323 entity to indicate its desire to set up a connection to the called entity.

Follow Table 3-15/Q.931 as modified in Table 13.

Table 13/H.225.0 – Setup

Information element	H.225.0 status (M/F/O/CM)	Length in H.225.0
Protocol discriminator	M	1
Call reference	M (Note 2)	3
Message type	M	1
Sending complete	O	1
Repeat indicator	F	NA
Bearer capability	M	5-6
Extended facility	O	8-*
Channel identification	FFS	NA
Facility	O	8-*
Progress indicator	O	2-4
Network specific facilities	F	NA
Notification indicator	O	2-*
Display	O	2-82
Keypad facility	O	2-34
Signal	O	2-3
Calling party number	O	2-131
Calling party subaddress	CM (Note 1)	NA
Called party number	O	2-131
Called party subaddress	CM (Note 1)	NA

Table 13/H.225.0 – Setup

Information element	H.225.0 status (M/F/O/CM)	Length in H.225.0
Redirecting Number	O	2-*
Transit network selection	F	NA
Low layer compatibility	FFS	NA
High layer compatibility	FFS	NA
User-user	M	2-131
NOTE 1 – Subaddresses are needed for some SCN call scenarios; they should not be used for packet-based network side only calls. NOTE 2 – If an ARQ was previously sent, the CRV used here shall be the same.		

The User-user information element contains the **Setup-UUIE** defined in the H.225.0 Message Syntax. The **Setup-UUIE** includes the following:

protocolIdentifier – Set to the version of H.225.0 supported.

h245Address – This is a specific transport address on which the calling endpoint or gatekeeper handling the call would like to establish the H.245 signalling. This should only be provided by the sender if it is capable of handling H.245 procedures before receiving a Connect on the Call Signalling channel.

sourceAddress – Contains the alias addresses of the source. The primary address shall be first. Note that the E.164 number of the source, if any, shall be contained within the Calling Party Number information element.

sourceInfo – Contains an **EndpointType** to allow the called party to determine whether the call involves a gateway or not.

destinationAddress – This is the address to which the endpoint wishes to be connected. The primary address shall be first. When calling an endpoint using only a dialled digit string, this address shall be placed in the H.225.0 call signalling message Called Party Number IE. The **destinationAddress**, if available, shall be included in the Setup message by terminals compliant with version 2 or higher of this Recommendation.

destCallSignalAddress – Needed to inform the gatekeeper of the destination terminal's call signalling transport address; redundant in the direct terminal-to-terminal case. In all cases where the information is available to the sender of the Setup message, this field shall be filled in.

destExtraCallInfo – Needed to make possible additional channel calls, i.e., for a 2 × 64 kbit/s call on the SCN side. Shall only contain dialled digit strings, E.164 numbers, or Private numbers and shall not contain the number of the initial channel. (See Note.)

destExtraCRV – CRVs for the additional SCN calls specified by **destExtraCallInfo.** Their use is for further study. They can be used to associate RAS signalling with the modified Q.931 signalling used in this Recommendation.

activeMC – Indicates that the calling endpoint is under the influence of an active MC.

conferenceID – Unique conference identifier.

conferenceGoal:

- **create** – Start a new conference.
- **invite** – Invite a party into an existing conference.
- **join** – Join an existing conference.

- **capability-negotiation** – Negotiate capabilities for a later loosely coupled conference.
- **callIndependentSupplementaryService** – Transport of supplementary services APDUs in a non-call related manner.

callServices – Provides information on support of optional Q-series protocols to gatekeeper and called terminal.

callType – Using this value, called party's gatekeeper can attempt to determine "real" bandwidth usage. The default value is **pointToPoint** for all calls; it should be recognized that the call type may change dynamically during the call and that the final call type may not be known when the Setup is sent.

sourceCallSignalAddress – Contains the transport address for the source; this value shall be used in the ARQ message by the receiver of the Setup. In all cases where the information is available to the sender of the Setup message, this field shall be filled in. The value of **sourceCallSignalAddress** shall be equal to the value that was used in the ARQ by the sender of the Setup, and shall be echoed by the endpoint receiving the Setup in its ARQ.

remoteExtensionAddress – Contains the alias address of a called endpoint in cases where this information is needed to traverse multiple Gateways. In all cases where the information is available to the sender of the Setup message, this field shall be filled in.

callIdentifier – A globally unique call identifier set by the originating endpoint which can be used to associate RAS signalling with the modified Q.931 signalling used in this Recommendation.

h245SecurityCapability – A set of capabilities the sender can use to secure the H.245 channel.

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

fastStart – Used only in the fast connect procedure, **fastStart** supports the signalling needed to open a logical channel. This uses the **OpenLogicalChannel** structure defined in ITU-T Rec. H.245, but the sender of this indicates the modes it prefers to receive and transmit, and the transport addresses where it expects to receive media streams.

mediaWaitForConnect – If TRUE, indicates that the recipient of the Setup message shall not transmit media until sending the Connect message.

canOverlapSend – If TRUE, indicates that the sender of Setup shall support overlap sending.

endpointIdentifier – This is an endpoint identifier that was assigned to the terminal in the RCF message. This field shall be present when the Setup is sent towards the gatekeeper where the endpoint is registered, and shall not be present when the Setup is sent to any other entity.

multipleCalls – If TRUE, this indicates that the sender of the message is capable of signalling multiple calls over a single call signalling connection.

maintainConnection – If TRUE, this indicates that the sender of the message is capable of supporting a signalling connection when no calls are currently signalled over the connection.

ConnectionParameters – Allow specification of parameters needed by gateways that provide multiple connection types and/or aggregation (for example, an H.323/H.320 gateway):

- **scnConnectionType** – Provides information to a gateway on the type of individual connection used to produce the entire SCN call. Endpoints or gatekeepers should fill in this field if the information is available to them. If the option "multirate" is indicated, then the information transfer rate octet in the bearer capability shall also indicate "multirate" and the rate multiplier octet shall indicate the number of connections. In all other cases, if the **scnConnectionType** field is present, it overrides any indication about the individual connection type contained in the transfer rate (octet #4) and rate multiplier (octet #4.1) of the bearer capability IE.

- **numberOfSCNConnections** – Indicates the number of connections of type **scnConnectionType** which are aggregated together to produce the SCN call. This field, when multiplied by the bandwidth of the individual connection specified in **scnConnectionType**, denotes the bandwidth for the entire call on the SCN. Endpoints or gatekeepers should fill in this field if the information is available to them. Note that if the **scnConnectionType** is set to unknown, then a unit of bandwidth of 64 kbit/s is assumed. If both this field and the **scnConnectionType** fields are present, then the total bandwidth indicated shall agree with the total SCN bandwidth indicated by the transfer rate (octet #4) and rate multiplier (octet #4.1) of the Bearer capability IE.
- **scnConnectionAggregation** – Indicates how the individual connections are aggregated together to produce the complete SCN call. Endpoints or gatekeepers should fill in this field if the information is available to them. The default option, to be used when the actual aggregation mechanism is unknown, is "auto". Where bonding is known to be used, but the precise bonding mode is unknown, then the option "bonded-mode1" should be used.

language – Indicates the language(s) in which the user would prefer to receive announcements and prompts. The field contains one or more RFC 1766 compliant language tags.

presentationIndicator – Indicates whether presentation of the **sourceAddress** should be allowed or restricted. If both **presentationIndicator** and the presentation indicator of the Calling Party Number IE are present and are in conflict, the presentation indicator of the Calling Party Number IE shall be used.

screeningIndicator – Indicates whether the **sourceAddress** was provided by the endpoint or network (gatekeeper), and whether the sourceAddress was screened by a gatekeeper. If both **screeningIndicator** and the screening indicator of the Calling Party Number IE are present and are in conflict, the screening indicator of the Calling Party Number IE shall be used.

serviceControl – Contains service-specific data, or references to it, that may be used as part of the setup procedure at the called endpoint (e.g., an image or icon to be displayed while alerting) as described, for example, in Annex K/H.323.

symmetricOperationRequired – If present, indicates that the called endpoint must select identical transmit and receive audio capabilities. This element shall not be included unless the **fastStart** element is also included.

capacity – This field indicates the sending endpoint's available call capacity at this point in time, assuming that this Setup message represents an active call. When sending this field, the endpoint shall include the **currentCallCapacity** element.

circuitInfo – This field provides information about the SCN circuit or circuits used for this call.

desiredProtocols – Identifies the type of protocols, in order of preference, the originating endpoint desires for its call (e.g., voice or fax). A resolving entity may use this field to locate an endpoint that also supports the protocol, giving consideration to the order of preference.

neededFeatures – This field specifies a list of generic features that are required in order for the call to succeed.

desiredFeatures – This field specifies a list of generic features that are preferred for the call, but are not required in order for it to succeed.

supportedFeatures – This field specifies a list of generic features that the sender support and has chosen to declare.

parallelH245Control – This field carries a sequence of tunnelled H.245 Terminal Capability Set PDUs and optionally Master Slave Determination PDUs. Each octet string shall contain exactly one H.245 PDU.

additionalSourceAddresses – This field carries a sequence of alias addresses that correspond to the second and subsequent Calling Party Number IEs in non-H.323 networks. For example, in ISDN, multiple calling party numbers may be present to support the "Two Calling party number information elements delivery option" defined in Annex A/Q.951.

hopCount – This field specifies an integer value to indicate the number of hops the call signalling can traverse further.

NOTE – If the **destExtraCallInfo** is present, a CRV for each call to be made may be supplied in **destExtraCRV**. These CRVs will be used to identify any response to each call launched. These procedures are for further study. If the **destExtraCRV** field is not present, a gateway shall aggregate all call information into a single response, with the effect that if one call fails on the SCN side, the entire call is treated as a failure.

7.3.11 Setup acknowledge

This message may be sent by an H.323 entity. However, it may be forwarded from the network via a gateway. Processing on receipt is optional, but an entity that indicates **canOverlapSend** in Setup shall support Setup Acknowledge.

The contents and semantics of a Setup Acknowledge message received from the network are defined in Table 3-16/Q.931, as modified in Table 14.

Table 14/H.225.0 – Setup acknowledge

Information element	H.225.0 status (M/F/O)	Length in H.225.0
Protocol discriminator	M	1
Call reference	M	3
Message type	M	1
Channel identification	FFS	NA
Display	O	2-82
User-user	M	2-131

For backward compatibility with systems prior to H.225.0 version 4, the sender of this message shall not include the **h4501SupplementaryService** or the **h245Control** field in the **h323-message-body** field of the User-user information element.

The User-user information element contains the **SetupAcknowledge-UUIE** defined in the H.225.0 Message Syntax. The **SetupAcknowledge-UUIE** includes the following:

protocolIdentifier – Set to the version of H.225.0 supported.

callIdentifier – A globally unique call identifier set by the originating endpoint, which can be used to associate RAS signalling with the modified Q.931 signalling used in this Recommendation.

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

7.3.12 Status

The Status message shall be used to respond to an unknown call signalling message or to a Status Inquiry message.

Follow Table 3-17/Q.931 as modified in Table 15.

Table 15/H.225.0 – Status

Information element	H.225.0 status (M/F/O)	Length in H.225.0
Protocol discriminator	M	1
Call reference (Note)	M	3
Message type	M	1
Cause	M	4-32
Call State	M	3
Display	O	2-82
User-user	M	2-131
NOTE – This message may carry the global call reference if the message applies to all calls on a connection carrying multiple calls.		

For backward compatibility with systems prior to H.225.0 version 4, the sender of this message shall not include the **h4501SupplementaryService** or the **h245Control** field in the **h323-message-body** field of the User-user information element.

The User-user information element contains the **Status-UUIE** defined in the H.225.0 Message Syntax. The **Status-UUIE** includes the following:

protocolIdentifier – Set to the version of H.225.0 supported.

callIdentifier – A globally unique call identifier set by the originating endpoint, which can be used to associate RAS signalling with the modified Q.931 signalling used in this Recommendation.

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

7.3.13 Status inquiry

The Status Inquiry message may be used to request call status as described in 8.4.2/H.323.

Follow Table 3-18/Q.931 as modified by Table 16.

Table 16/H.225.0 – Status inquiry

Information element	H.225.0 status (M/F/O)	Length in H.225.0
Protocol discriminator	M	1
Call reference (Note)	M	3
Message type	M	1
Display	O	2-82
User-user	M	2-131
NOTE – This message may carry the global call reference if the message applies to all calls on a connection carrying multiple calls.		

For backward compatibility with systems prior to H.225.0 version 4, the sender of this message shall not include the **h4501SupplementaryService** or the **h245Control** field in the **h323-message-body** field of the User-user information element.

The User-user information element contains the **StatusInquiry-UUIE** defined in the H.225.0 Message Syntax. The **StatusInquiry-UUIE** includes the following:

protocolIdentifier – Set to the version of H.225.0 supported.

callIdentifier – A globally unique call identifier set by the originating endpoint which can be used to associate RAS signalling with the modified Q.931 signalling used in this Recommendation.

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

7.4 Q.932-based H.225.0 call signalling message details

The messages defined in the following are derived from ITU-T Recs Q.932 and H.450. Refer to ITU-T Recs Q.932 and H.450 for further details.

7.4.1 Facility

The Facility message shall be used to provide information on where a call should be directed (**FacilityReason** = **routeCallToMC**), or for an endpoint to indicate that the incoming call must go through a gatekeeper (**FacilityReason** = **routeCallToGatekeeper**).

In order to signal call redirection specific to H.323 procedures, the User-user information element of the Facility message is used. This particular case shall be indicated by coding a Facility IE of length zero. In this case, the Facility information element shall consist of exactly 2 octets. An H.323 entity shall handle the empty (H.323-specific) Facility IE properly and shall be capable of skipping other Facility IEs that it does not understand.

The Facility message may be used to request or acknowledge a supplementary service according to H.450.x-series Recommendations. For that reason, one or more H.450 Supplementary Service APDUs shall be carried within the User-user information element of the Facility message. The H.450 Supplementary Service APDUs shall be coded according to clause 8/H.450.1. The Facility information element shall be contained with length zero. Note that a Facility message of H.225.0 version 2 or version 3 that carries only H.450 Supplementary Service APDUs might choose not to include the Facility-UUIE, but instead use the "**empty**" **h323-message-body** choice. In this case, a Facility message would not have a **callIdentifier** field in it. In H.225.0 version 4 and higher, a sender shall include a Facility-UUIE carrying a **callIdentifier** field in every call-associated Facility message, and shall set the **reason** field value to **transportedInformation**.

If a Facility IE carrying semantics of ITU-T Rec. Q.932 and encoded as defined in ITU-T Rec. Q.932 and ITU-T Rec. Q.95.x is present, it shall consist of at least 8 octets as required by Table 7-2/Q.932. The use of Facility IEs of that type is for further study.

The Facility message may be used by an endpoint or gatekeeper to request the recipient to establish an H.245 channel between the two entities (**FacilityReason** = **startH245**).

The Facility message may be used by an endpoint or gatekeeper to send a new set of tokens in the **tokens** and/or **cryptoTokens** field of the Facility message (**FacilityReason** = **newTokens**). This may be useful, for example, for applications in which tokens are used to allow some action to take place only for a limited amount of time.

Follow 7.1.1/Q.932 and 10.8 of ISO/IEC 11582, as modified in Table 17.

Table 17/H.225.0 – Facility

Information element	H.225.0 status (M/F/O)	Length in H.225.0
Protocol discriminator	M	1
Call reference (Note 1)	M	3
Message type	M	1
Extended facility	O (Note 2)	8-*
Facility	O (Note 2)	2 or 8-*
Notification indicator	O	2-*
Display	O	2-82
Calling Party Number	F	NA
Called Party Number	F	NA
User-user	M	2-131
NOTE 1 – This message may carry the global call reference if the message applies to all calls on a connection carrying multiple calls. NOTE 2 – If the Facility message is used for carrying Q.95.x supplementary service signalling, one of either the Facility or Extended Facility information elements is required. If the Facility message is used for Supplementary Service control according to H.450.x-series Recommendations, or if the Facility message is used for the reroute to MC/GK functions, then the zero-length Facility information element is required.		

Coding of Message Type information element

The message type information element of the Facility message shall be coded "0110 0010".

The User-user information element contains the Facility-UUIE defined in the H.225.0 Message Syntax. The Facility-UUIE includes the following:

protocolIdentifier – Set to the version of H.225.0 supported.

alternativeAddress – This is a specific transport address to which the calling party should direct the call; if present, **alternativeAliasAddress** is not needed.

alternativeAliasAddress – Contains aliases that can be used to redirect the call; if an alias is provided, **alternativeAddress** is not needed.

conferenceID – Unique conference identifier; not needed if the **conferences** field is used.

reason – More information about the Facility message. A **reason** of **featureSetUpdate** indicates that the purpose of the message is to update **featureSet** information that was sent previously. A **reason** of **forwardedElements** indicates that the purpose of the message is to forward elements of another message in case that message cannot be sent, as would be the case when a routing gatekeeper receives a Call Proceeding message after it has already sent Call Proceeding. A **reason** of **transportedInformation** indicates that the purpose of the message is to transport higher-layer information, for example in the **h4501SupplementaryService** field; the **Facility-UUIE** in this case is included only in order to provide the **callIdentifier**.

callIdentifier – A globally unique call identifier set by the originating endpoint which can be used to associate RAS signalling with the modified Q.931 signalling used in this Recommendation.

destExtraCallInfo – Needed to make possible additional channel calls, i.e., for a 2 × 64 kbit/s call on the SCN side. Shall only contain dialled digit strings, E.164 numbers, or Private numbers and shall not contain the number of the initial channel.

remoteExtensionAddress – Contains the alias address of a called endpoint in cases where this information is needed to traverse multiple Gateways.

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

conferences – One or more conferences that may be joined.

h245Address – This is a specific transport address on which the endpoint or gatekeeper sending this facility would like the recipient to establish H.245 signalling. Note that this field may be present when an intermediate signalling entity is conveying the **h245Address** field from a Call Proceeding message. The receiving entity is instructed to initiate H.245 procedures only when **reason** is **startH245**.

fastStart – Used only in the fast connect procedure, **fastStart** supports the signalling needed to open a logical channel. This uses the **OpenLogicalChannel** structure defined in ITU-T Rec. H.245, but the sender of this indicates the modes it prefers to receive and transmit, and the transport addresses where it expects to receive media streams. This field is present in a Facility message when a routing gatekeeper received it in a Call Proceeding message from the called user and is forwarding the information to the calling user. This field shall not be included by an endpoint.

multipleCalls – If TRUE, this indicates that the sender of the message is capable of signalling multiple calls over a single call signalling connection.

maintainConnection – If TRUE, this indicates that the sender of the message is capable of supporting a signalling connection when no calls are currently signalled over the connection.

fastConnectRefused – A called endpoint should return this element in any message up to and including the Connect message when establishing a call to indicate that it refuses the Fast Connect procedure. This field is present in a Facility message when a routing gatekeeper received it in a Call Proceeding message from the called user and is forwarding the information to the calling user.

serviceControl – Contains service-specific data, or references to it, that could be used by an endpoint or gateway (e.g., for displaying a menu of options to a participant in a call), as described, for example, in Annex K/H.323.

circuitInfo – This field provides information about the SCN circuit or circuits used for this call.

featureSet – This field specifies a set of generic features that relate to this call.

destinationInfo – Contains an **EndpointType** to allow the caller to determine whether the call involves a gateway or not. This field is present in a Facility message when a routing gatekeeper received it in a Call Proceeding message from the called user and is forwarding the information to the calling user. This field did not exist in the Facility message prior to H.225.0 version 4.

h245SecurityMode – An H.323 entity that receives a Setup message with the **h245SecurityCapability** set responds with the corresponding, acceptable **h245SecurityMode** in the Call Proceeding, Alerting, Progress, or Connect. This field is present in a Facility message when a routing gatekeeper received it in a Call Proceeding message from the called user and is forwarding the information to the calling user. This field did not exist in the Facility message prior to H.225.0 version 4.

7.4.2 Notify

This message may be sent by an H.323 entity. Processing on receipt is optional.

Follow Table 3-8/Q.931 as modified in Table 18.

Table 18/H.225.0 – Notify

Information element	H.225.0 status (M/F/O)	Length in H.225.0
Protocol discriminator	M	1
Call reference	M	3
Message type	M	1
Bearer capability	O (Note)	5-6
Notification indicator	M	3
Display	O	2-82
User-user	M	2-131
NOTE – Included to indicate a change of the bearer capability.		

For backward compatibility with systems prior to H.225.0 version 4, the sender of this message shall not include the **h4501SupplementaryService** or the **h245Control** field in the **h323-message-body** field of the User-user information element.

The User-user information element contains the **Notify-UUIE** defined in the H.225.0 Message Syntax. The **Notify-UUIE** includes the following:

protocolIdentifier – Set to the version of H.225.0 supported.

callIdentifier – A globally unique call identifier set by the originating endpoint which can be used to associate RAS signalling with the modified Q.931 signalling used in this Recommendation.

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

7.4.3 Other messages

The call control messages which can carry optional Facility, Extended Facility, or Notification Indicator information elements are specified in 8.3.

7.5 H.225.0 call signalling timer values

The following Q.931 timers shall be supported:

- The "setup timer" T303 (see Tables 9-1/Q.931 and 9-2/Q.931) defining how long the calling endpoint shall wait for an Alerting, Call Proceeding, Connect, Release Complete or other message from the called endpoint after it has sent a Setup message. This time-out value shall be at least 4 seconds. Note that some applications may appear in networks which have inherently longer delays (for example, compare the Internet to a local enterprise network or intranet).
- The "establishment timer" T301 (see Tables 9-1/Q.931 and 9-2/Q.931) defining after which time the calling endpoint shall stop waiting for the called endpoint to respond. This timer starts when Alerting is received and normally terminates on Connect or when the caller terminates the call attempt and sends Release Complete. This time-out value shall be 180 seconds (3 minutes) or greater.
- The "overlap sending timer" T302 (see Tables 9-1/Q.931 and 9-2/Q.931) defining after which time the called endpoint shall stop waiting for the dialled digits from calling endpoint

while in the overlap sending. This timer starts when SETUP ACK is sent or INFORMATION received and normally terminates when sending complete indication is received. This timeout value shall be 10-15 seconds.

- The "overlap receiving timer" T304 (see Tables 9-1/Q.931 and 9-2/Q.931) defining after which time the calling endpoint shall stop waiting for the dialled digits from called endpoint user while in the overlap receiving. This timer starts when SETUP ACK is received, restarts when INFORMATION is sent, and normally terminates CALL PROCEEDING, ALERTING or CONNECT is received. This timeout value shall be at least 20 seconds.
- The "incoming call proceeding timer" T310 (see Tables 9-1/Q.931 and 9-2/Q.931) defining after which time the called endpoint shall stop waiting for the dialled digits from calling endpoint while in the overlap sending. This timer starts when CALL PROCEEDING is received and normally terminates on ALERTING, CONNECT or when the caller terminates the call attempt and sends Release Complete. This timeout value shall be at least 10 seconds.
- The "status timer" T322 (see Tables 9-1/Q.931 and 9-2/Q.931) defining after which time the called endpoint shall stop waiting STATUS message response to STATUS ENQUIRY it has sent. This timer starts when STATUS ENQUIRY is sent normally terminates when STATUS message is received. This timeout value shall be at least 4 seconds.

Note that the packet-based network-side values of these timers is the same as that used in the SCN.

Other timers may be supported as part of optional H.450.x-series Supplementary Service Recommendations.

7.6 H.225.0 common message elements

This clause describes ASN.1 structures that are used in more than one Registration, Admission, and Status (RAS) messages. Some may also be used in the User-user part of the call signalling messages.

requestSeqNum in messages is used to keep track of multiple outstanding requests. Any associated response messages (success or failure) shall have the corresponding **requestSeqNum** returned with it. Retransmitted messages shall have the same **requestSeqNum**. **RequestSeqNum** increments by 1 modulo 65536.

The **protocolIdentifier** is included as part of discovery, registration and Setup/Connect to allow the parties involved to determine the vintage of the implementations involved.

nonStandardParameter – This parameter is optional in the discovery, registration, and Setup/Connect sequences to allow the parties involved to determine the non-standard status of the endpoints involved. A gatekeeper or gateway is not obligated to pass on **nonStandardData** it does not support or understand as this might interfere with its operations.

The **TransportAddress** structure is meant to capture the various transport formats and includes any transport-specific scheme in addition to the possibly local reference to a TSAP identifier.

IPv4 and IPv6 addresses shall be encoded with the most significant octet of the address being the first octet in the respective OCTET STRING, e.g., the class B IPv4 address 130.1.2.97 shall have the "130" being encoded in the first octet of the OCTET STRING, followed by the "1" and so forth.

The IPv6 address a148:2:3:4:a:b:c:d shall have the "a1" encoded in the first octet, "48" in the second, "00" in the third, "02" in the fourth and so forth.

A **TransportAddress** of type **ipSourceRoute** in which the **route** SEQUENCE has no entries shall be interpreted as representing the same address as of type **ipAddress** which contains the same values for both **ip** and **port**.

IPX addresses, **node, netnum**, and **port** shall be encoded with the most significant octet of each field being the first octet in the respective OCTET STRING.

Note that this structure does not use the Transport Address = "packet-based network Address plus TSAP identifier" language of ITU-T Rec. H.323. Instead, the terms common in each transport domain are used.

The **EndpointType** structure conveys information about the H.323 entity at the end of the signalling link. The H.323 entity would complete one or more of the **gatekeeper**, **gateway**, **mcu**, or **terminal** message elements. If the H.323 entity has an MC, then the **mc** Boolean would be TRUE. Clause 6.3/H.323 describes the representation of an MCU when collocated with a gateway; in this case, the H.323 device may include both the **gateway** and the **mcu** elements within its **EndpointType** definition. Presence of the **set** component indicates that the entity is a Simple Endpoint Type (SET) device as defined, for example, in Annex F/H.323. The bit positions in the **set** component indicate the type of SET device; their meaning is defined in Annex F/H.323 and other Recommendations that specify SET device types. The **supportedTunnelledProtocols** field supplies a prioritized list (highest priority first) of supported tunnelled protocols.

The **TunnelledProtocol** structure identifies a tunnelled signalling protocol as described, for example, in Annexes M.1 and M.2/H.323. The **tunnelledProtocolObjectID** field is an **OBJECT IDENTIFIER** identifying the protocol being tunnelled. The **tunnelledProtocolAlternateID** provides an alternate identifier format. The **subIdentifier** field allows specification of a particular version of a standard protocol.

The **TunnelledProtocolAlternateIdentifier** structure provides a string-based identifier format for a tunnelled protocol. The **protocolType** provides the general type of protocol, such as ISUP. The **protocolVariant** field provides a specific variation of that standard, such as ANSI.

Tunnelled protocols that are defined as of this Recommendation are shown in Tables VI.1 and VI.2. Note that tunnelling is not restricted to the protocols listed in those tables.

The **GatewayInfo** structure contains a **protocol** element, which allows the gateway to indicate the protocols it supports.

The **SupportedProtocols** structure indicates a choice of protocols with which an H.323 entity has the capability to interwork. For example, selection of the **h310** choice indicates that the entity provides interworking with H.310.

In each supported protocol capability structure (**H310Caps**, **H320Caps**, etc.), the **dataRatesSupported** element indicates the data rates supported for each protocol the device supports. The **supportedPrefixes** element indicates the prefixes associated with a supported protocol, and in some cases also with the data rates.

The **McuInfo** structure contains a **protocol** element, which allows the MCU to indicate the protocols it supports.

The **CapacityReportingCapability** structure indicates an endpoint's ability to report call capacity information.

The **CapacityReportingSpecification** structure indicates the call capacity information that an endpoint is requested to report. **callStart** indicates a request for capacity information at the beginning of the call (i.e., in the ARQ or Setup). **callEnd** indicates a request for capacity information at the end of the call (i.e., in the DRQ or Release Complete). An empty **when** sequence indicates a request that the endpoint not report capacity information.

The **CallCapacityInfo** structure allows an endpoint to indicate its call acceptance capacity for each type of call the endpoint supports. It therefore represents the current idle status of the endpoint. For example, in a voice gateway **CallCapacityInfo** would represent the number of idle circuits.

The **CallCapacity** structure allows an endpoint to indicate its maximum capacity for each type of call and its current available capacity for each type of the call the endpoint supports.

The **CallsAvailable** structure represents a subset of an endpoint's total call capacity. The **group** field allows the subset to be identified by a group label. The **group** may be the same as that reported in the **CircuitIdentifier**.

The **DataRate** structure provides gateway protocol rate information. **channelRate** is the basic channel rate in hundreds of bits. **channelMultiplier** indicates the number of channels at the channelRate. For example, if a gateway supports a 3B call, **channelMultiplier** = 3 and **channelRate** = 640 for a 64 kbit/s channel.

The **VendorIdentifier** structure allows a vendor to identify a product. The **vendor** element allows identification in terms of country code, extension, and manufacturer code. **productId** and **versionId** are text strings that can provide product information. The **enterpriseNumber** field identifies the manufacturer and is assigned by the Internet Assigned Numbers Authority (IANA).

The **H221NonStandard** structure allows definition of a nonstandard field. The **t35CountryCode** element shall identify the country, as described in Annex A/T.35. The **t35Extension** element shall contain a country code extension that is assigned nationally, unless the **t35CountryCode** is binary "1111 1111", in which case this field shall contain the country code found in Annex B/T.35. The **manufacturerCode** shall be assigned nationally and identifies an equipment manufacturer.

The **AliasAddress** structure is meant to capture the various external address formats that reference a particular transport location on the packet-based network. When registering an address consisting of dialled digits with a gatekeeper, an endpoint shall use the **dialledDigits** field and shall use only the digits 0-9. When registering an E.164 address with a gatekeeper, an endpoint shall use the **e164Number** field and shall use only the digits 0-9. When registering or otherwise representing a prefix, an endpoint shall use the **dialledDigits** field and shall use only the digits 0-9 and "#" and "*". The **mobileUIM** field is an identification module for systems compatible with 2nd Generation and 3rd Generation wireless networks, and permits interworking with Public Land Mobile Networks as described, for example, in Annex E/H.246.

The **AddressPattern** structure allows specification of a wildcarded **AliasAddress** or a range of **PartyNumber**s. The **wildcard** field represents the possible wildcarded expansion of the **AliasAddress** structure. For dialled digits or E.164 numbers this expansion is possible at the end of the number. For email addresses the expansion is possible at the beginning. For example, if wildcard is "+1 303", the pattern could represent any number in the Denver area code. The **range** field of the **AddressPattern** structure represents a range of addresses, including the indicated start and end of range.

The mechanisms that an endpoint uses to determine the address type is left as an implementation issue. The representation of the various number types in messages is captured in Table 19. Note that if an endpoint does not know the type or scope of an address, then it should represent this as Private Unknown when coded in H.225.0 call signalling messages and as a **dialledDigits AliasAddress** when coded in RAS messages.

Table 19/H.225.0 – Type of number representation mapping

Type of number	Q.931 representation	H.225.0 information element representation	H.225.0 UUIE representation
Unknown (default and version 1 interoperability mode)	Private Numbering Plan, Type of number = Unknown ("000") (Note 1)	Private Numbering Plan, Type of number = Unknown ("000")	**dialledDigits AliasAddress** (Note 2)
Private unknown	Private Numbering Plan, Type of number = Unknown ("000") (Note 1)	Private Numbering Plan, Type of number = Unknown ("000") (Note 1)	**dialledDigits AliasAddress** (Note 2)
Private, Level 2 Regional Number	Private Numbering Plan, Type of number = Level 2 Regional Number ("001")	Private Numbering Plan, Type of number = Unknown ("000") (Note 1)	**privateNumber** of **PartyNumber AliasAddress**, **TypeOfNumber** = **level2RegionalNumber**
Private, Level 1 Regional Number	Private Numbering Plan, Type of number = Level 1 Regional Number ("010")	Private Numbering Plan, Type of number = Unknown ("000") (Note 1)	**privateNumber** of **PartyNumber AliasAddress**, **TypeOfNumber** = **level1RegionalNumber**
Private, PISN specific Number	Private Numbering Plan, Type of number = PISN specific Number ("011")	Private Numbering Plan, Type of number = Unknown ("000") (Note 1)	**privateNumber** of **PartyNumber AliasAddress**, **TypeOfNumber** = **pISNSpecificNumber**
Private, Level 0 Regional Number (Local)	Private Numbering Plan, Type of number = Level 0 Regional Number ("100")	Private Numbering Plan, Type of number = Unknown ("000") (Note 1)	**privateNumber** of **PartyNumber AliasAddress**, **TypeOfNumber** = **localNumber**
E.164 Public number, unknown	ISDN/Telephony Numbering Plan, Type of number = Unknown ("000")	ISDN/Telephony Numbering Plan, Type of number = Unknown ("000")	**e164Number** of **PartyNumber AliasAddress**, **TypeOfNumber** = **Unknown**
E.164 Public number, International Number	ISDN/Telephony Numbering Plan, Type of number = International Number ("001")	ISDN/Telephony Numbering Plan, Type of number = International Number ("001")	**e164Number** of **PartyNumber AliasAddress**, **TypeOfNumber** = **internationalNumber**
E.164 Public number, National Number	ISDN/Telephony Numbering Plan, Type of number = National Number ("010")	ISDN/Telephony Numbering Plan, Type of number = National Number ("010")	**e164Number** of **PartyNumber AliasAddress**, **TypeOfNumber** = **nationalNumber**

Table 19/H.225.0 – Type of number representation mapping

Type of number	Q.931 representation	H.225.0 information element representation	H.225.0 UUIE representation
E.164 Public number, Network Specific Number	ISDN/Telephony Numbering Plan, Type of number = NetworkSpecific Number ("011")	ISDN/Telephony Numbering Plan, Type of number = NetworkSpecificNumber ("011")	**e164Number** of **PartyNumber AliasAddress**, **TypeOfNumber** = **networkSpecific Number**
E.164 Public number, Subscriber Number	ISDN/Telephony Numbering Plan, Type of number = Subscriber Number ("100")	ISDN/Telephony Numbering Plan, Type of number = Subscriber Number ("100")	**e164Number** of **PartyNumber AliasAddress**, **TypeOfNumber** = **subscriberNumber**
E.164 Public number, Abbreviated Number	ISDN/Telephony Numbering Plan, Type of number = Abbreviated Number ("110")	ISDN/Telephony Numbering Plan, Type of number = Abbreviated Number ("110")	**e164Number** of **PartyNumber AliasAddress**, **TypeOfNumber** = **abbreviatedNumber**
NOTE 1 – When Numbering plan identification = Private, the private number digits are encoded in **privateNumber** of **PartyNumber**, which includes the type of number. The Type of number field in the information element shall be ignored on reception, and coded according to this table on transmission. NOTE 2 – A **privateTypeOfNumber** = **Unknown PartyNumber AliasAddress** shall be treated the same as a **dialledDigits AliasAddress**.			

The **MobileUIM** structure represents an identification module for systems compatible with 2nd Generation and 3rd Generation wireless networks. The choices available are:

- **ansi-41-uim** – This is for wireless networks defined by American standards.
- **gsm-uim** – This is for wireless networks defined by European standards.

The **ANSI-41-UIM** structure identifies an identification module for systems compliant with American standards for wireless networks. The choices available are:

- **imsi** – This is for International Mobile Station Identification numbers.
- **min** – This is for Mobile Identification Numbers.
- **mdn** – This is for Mobile Directory Numbers.
- **msisdn** – This is for Mobile Station ISDN numbers.
- **esn** – This is for Electronic Serial Numbers.
- **mscid** – This is for Mobile Switching Center numbers plus Market Identification or System Identification numbers.
- **sid** – This is for System Identification numbers.
- **mid** – This is for Market Identification numbers.
- **systemMyTypeCode** – This is for vendor identification numbers.
- **systemAccessType** – This is for the system access type.
- **qualificationInformationCode** – This is for the qualification information code.
- **sesn** – This is for SIM Electronic Serial Numbers.
- **soc** – This is for System Operator Codes.

The **GSM-UIM** structure identifies an identification module for systems compliant with European standards for wireless networks. The choices available are:

- **imsi** – This is for International Mobile Station Identification.
- **tmsi** – This is for Temporary Mobile Station Identification.
- **msisdn** – This is for Mobile Station ISDN numbers.
- **imei** – This is for International Mobile Equipment Identification numbers.
- **hplmn** – This is for Home Public Land Mobile Network Numbers.
- **vplmn** – This is for Visiting Public Land Mobile Network Numbers.

The **ExtendedAliasAddress** structure provides a means for associating common information with alias addresses. The **presentationIndicator** indicates whether presentation of the **address** should be allowed or restricted. The **screeningIndicator** indicates whether the **address** was provided by the endpoint or by the network, and whether it has been screened by the network.

The **Endpoint** structure is used to indicate back-up, redundant, or alternative information about an endpoint:

- **nonStandardData** – Carries information not defined in this Recommendation (for example, proprietary data).
- **aliasAddress** – This is a list of alias addresses, by which other endpoints may identify this endpoint.
- **callSignalAddress** – This is the call signalling transport address for this endpoint.
- **rasAddress** – This is the registration and status transport address for this endpoint.
- **endpointType** – This specifies the type of the endpoint.
- **tokens** – Tokens associated with this endpoint (i.e., endpoint described in the **Endpoint** structure).
- **cryptoTokens** – **CryptoTokens** associated with this endpoint (i.e., the endpoint described in the **Endpoint** structure).
- **priority** – Used when a SEQUENCE of **Endpoints** is presented. Endpoints with lower priority numbers are preferred over endpoints with higher priority numbers. Endpoints without priority numbers are equivalent to those with a priority of 0 (highest priority).
- **remoteExtensionAddress** – Contains the alias address of an endpoint in cases where this information is needed to traverse multiple gateways.
- **destExtraCallInfo** – Contains external addresses for multiple calls.
- **alternateTransportAddresses** – Indicates support for transports other than TCP.

The **alternateTransportAddresses** structure conveys call signalling addresses for transports other than TCP.

The **UseSpecifiedTransport** structure defines a choice of signalling transport protocols. A value of **tcp** indicate the TCP protocol, a value of **annexE** indicates the protocol defined by Annex E/H.323, and a value of **sctp** indicates the usage of the Stream Control Transmission Protocol (SCTP).

The **AlternateGK** structure is used to indicate a list of alternative, or back-up, gatekeepers:

- **rasAddress** – The transport address used for RAS signalling.
- **gatekeeperIdentifier** – Optionally included to identify the back-up or alternative gatekeeper. If it is supplied, it shall be included in future RAS messages sent to the back-up gatekeeper.
- **needToRegister** – Set to TRUE to indicate that the endpoint must register with the alternate before sending other RAS requests.

- **priority** – Indicates the priority of the gatekeeper back-up or alternative. A lower number implies a higher priority.

The **AltGKInfo** structure is used to provide information about alternate gatekeepers:

- **alternateGatekeeper** – Sequence of prioritized alternate gatekeepers.
- **altGKisPermanent** – TRUE to indicate that all future RAS signals should be redirected to a gatekeeper listed in the **alternateGatekeeper** field; FALSE if only the message that caused the Reject should be redirected. This flag shall be set to TRUE if a **needToRegister** flag is set to TRUE in the **alternateGatekeeper** field.

The **QseriesOptions** structure supplies information to the gatekeeper or other endpoints concerning the support provided by a terminal for optional Q-series protocols. It is used in the ARQ, Setup, and GRQ messages. The usage of QSeriesOptions is not yet defined pending further study.

The **GloballyUniqueID** and **ConferenceIdentifier** are meant to be globally unique identifiers (**GloballyUniqueID**), the use of which is described in ITU-T Rec. H.323. A **GloballyUniqueID** is encoded with octet zero being encoded first. A **GloballyUniqueID** is formed according to Table 20.

Table 20/H.225.0 – Globally unique ID formation

Field	Data type	Octet No.	Note
time_low	Unsigned 32-bit integer	0-3	The low field of the timestamp
time_mid	Unsigned 16-bit integer	4-5	The middle field of the timestamp
time_hi_and_version	Unsigned 16-bit integer	6-7	The high field of the timestamp multiplexed with the version number
clock_seq_hi_and_reserved	Unsigned 8-bit integer	8	The high field of the clock sequence multiplexed with the variant
clock_seq_low	Unsigned 8-bit integer	9	The low field of the clock sequence
node	Unsigned 48-bit integer	10-15	The spatially unique node identifier

The **GloballyUniqueID** consists of a record of 16 octets and shall not contain padding between fields. The total size is 128 bits.

To minimize confusion about bit assignments within octets, the **GloballyUniqueID** record definition is defined only in terms of fields that are integral numbers of octets. The version number is multiplexed with the timestamp (*time_high*), and the variant field is multiplexed with the clock sequence (*clock_seq_high*).

The timestamp is a 60-bit value represented by Coordinated Universal Time (UTC) as a count of 100 nanosecond intervals since 00:00:00.00, 15 October 1582 (the date of Gregorian reform to the Christian calendar).

The version number is multiplexed in the 4 most significant bits of the *time_hi_and_version* field, and is set to 1 (binary "0001").

The variant field determines the layout of the **GloballyUniqueID**. The structure of a DCE **GloballyUniqueID** is fixed across different versions. Other **GloballyUniqueID** variants may not interoperate with a DCE **GloballyUniqueID**. Interoperability of **GloballyUniqueID**s is defined as the applicability of operations such as string conversion, comparison, and lexical ordering across different systems. The *variant* field consists of a variable number of the MSBs of the *clock_seq_hi_and_reserved* field (see Table 21).

Table 21/H.225.0 – Contents of the DCE variant field

msb1	msb2	msb3	Description
0	–	–	Reserved, NCS backward compatibility
1	0	–	DCE variant
1	1	0	Reserved, Microsoft Corporation GUID
1	1	1	Reserved for future definition

The clock sequence is required to detect potential losses of monotonicity of the clock. The clock sequence is encoded in the 6 least significant bits of the *clock_seq_hi_and_reserved* field and in the *clock_seq_low* field.

The *node* field consists of the IEEE address, usually the host address. For systems with multiple IEEE 802 nodes, any available node address can be used. The lowest addressed octet (octet number 10) contains the global/local bit and the unicast/multicast bit, and is the first octet of the address transmitted on an 802.3 packet-based network.

The clock sequence value should be changed whenever:

- the **GloballyUniqueID** generator detects that the local value of UTC has gone backward; this may be due to normal functioning of the DCE Time Service.
- the **GloballyUniqueID** generator has lost its state of the last value of UTC used, indicating that time may have gone backward; this is typically the case on reboot.

While a node is operational, the **GloballyUniqueID** generator always saves the last UTC used to create a **GloballyUniqueID**. Each time a new **GloballyUniqueID** is created, the current *UTC* is compared to the saved value and if either the current value is less (the non-monotonic clock case) or the saved value was lost, then the *clock sequence* is incremented modulo 16 384, thus avoiding production of duplicate **GloballyUniqueID**s.

The *clock sequence* should be initialized to a random number to minimize the correlation across systems.

A **GloballyUniqueID** is generated according to the following algorithm:

1) Determine the values for the UTC-based timestamp and clock sequence to be used in the **GloballyUniqueID**.

2) Set the *time_low* field equal to the least significant 32 bits (bits numbered 0 to 31 inclusive) of the timestamp in the same order of significance.

3) Set the *time_mid* field equal to the bits numbered 32 to 47 inclusive of the timestamp in the same order of significance.

4) Set the 12 least significant bits (bits numbered 0 to 11 inclusive) of the *time_hi_and_version* field equal to the bits numbered 48 to 59 inclusive of the timestamp in the same order of significance.

5) Set the 4 most significant bits (bits numbered 12 to 15 inclusive) of the *time_hi_and_version* field to the 4-bit version number corresponding to the **GloballyUniqueID** version being created, as shown in Table 21.

6) Set the *clock_seq_low* field to the 8 least significant bits (bits numbered 0 to 7 inclusive) of the *clock sequence* in the same order of significance.

7) Set the 6 least significant bits (bits numbered 0 to 5 inclusive) of the *clock_seq_hi_and_reserved* field to the 6 most significant bits (bits numbered 8 to 13 inclusive) of the *clock sequence* in the same order of significance.

8) Set the 2 most significant bits (bits numbered 6 and 7) of the *clock_seq_hi_and_reserved* to 0 and 1, respectively.

9) Set the *node* field to the 48-bit IEEE address in the same order of significance as the address.

If a system wants to generate a **GloballyUniqueID** but has no IEEE 802 compliant network card or other source of IEEE 802 addresses, then an alternative method should be used to generate a replacement value for the address. The ideal solution is to obtain a 47-bit cryptographic quality random number, and use it as the most significant 47 bits of the node ID, with the least significant bit of the first octet of the node ID set to 1. This bit is the unicast/multicast bit, which will never be set in IEEE 802 addresses obtained from network cards; hence, there can never be a conflict between **GloballyUniqueID**s generated by machines with and without network cards.

If a system does not have a primitive to generate cryptographic quality random numbers, then in most systems there are usually a fairly large number of sources of randomness available from which one can be generated. Such sources are system specific, but often include the percentage of memory in use, the size of main memory in bytes, the amount of free main memory in bytes, the size of the paging or swap file in bytes, free bytes of paging or swap file, the total size of user virtual address pace in bytes, the total available user address space bytes, the size of boot disk drive in bytes, the free disk space on boot drive in bytes, the current time, the amount of time since the system booted, the individual sizes of files in various system directories, etc.

For use in human-readable text, a **GloballyUniqueID** string representation is specified as a sequence of fields, some of which are separated by single dashes.

Each field is treated as an integer and has its value printed as a zero-filled hexadecimal digit string with the most significant digit first. The hexadecimal values a to f inclusive are output as lower case characters, and are case insensitive on input. The sequence is the same as the **GloballyUniqueID** constructed type.

The formal definition of the **GloballyUniqueID** string representation is provided by the following extended BNF:

```
UUID                     = <time_low> <hyphen> <time_mid> <hyphen>
                           <time_high_and_version> <hyphen>
                           <clock_seq_and_reserved>
                           <clock_seq_low> <hyphen> <node>
time_low                 = <hexOctet> <hexOctet> <hexOctet> <hexOctet>
time_mid                 = <hexOctet> <hexOctet>
time_high_and_version    = <hexOctet> <hexOctet>
clock_seq_and_reserved   = <hexOctet>
clock_seq_low            = <hexOctet>
node                     = <hexOctet><hexOctet><hexOctet>
                           <hexOctet><hexOctet><hexOctet>
hexOctet                 = <hexDigit> <hexDigit>p
hexDigit                 = <digit> | <a> | <b> | <c> | <d> | <e> | <f>
digit                    = "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" |
                           "8" | "9"
hyphen                   = "-"
a                        = "a" | "A"
b                        = "b" | "B"
c                        = "c" | "C"
d                        = "d" | "D"
e                        = "e" | "E"
f                        = "f" | "F"
```

The following is an example of the string representation of a **GloballyUniqueID**:

f81d4fae-7dec-11d0-a765-00a0c91e6bf6

timeToLive is a number of seconds that a registration is to be considered valid.

The **H248PackagesDescriptor** structure is an octet string, which will contain ASN.1 PER encoded H.248 **PackagesDescriptor**.

The **H248SignalsDescriptor** structure is an octet string, which will contain ASN.1 PER encoded H.248 **SignalsDescriptor**.

The **FeatureDescriptor** structure is a **GenericData** element that is used to generically identify a feature.

CircuitInfo – This structure provides information about the SCN circuit or circuits used for this call. The **sourceCircuitID** field provides information about the source circuit when the call originates on the SCN, and might be used by an ingress gateway to report the source circuit identifier to the gatekeeper. The **destinationCircuitID** provides information about the destination circuit when the call terminates on the SCN, and might be used by a gatekeeper to select a destination circuit on an egress gateway.

The **CircuitIdentifier** structure designates a facility for purposes of reporting by a gateway or selection by a gatekeeper. The **CircuitIdentifier** structure supports a variety of interfaces.

The **CicInfo** structure designates SS7 bearer channels. The **cic** field is the circuit identifier code as defined in ITU-T Rec. Q.763, encoded with the least significant bits in the first octet and the most significant bits in the last octet. The **pointCode** field contains the point code as defined in ITU-T Rec. Q.763. The first octet of the **pointCode** identifies the network (network indicator code) and the remaining octets identify the SS7 point code value. The **cic** and **pointCode** fields are variable in length to allow for national variants.

The **GroupID** structure identifies a physical or logical **group** and a **member** (or set of **member**s) within that group. For example, **group** could identify a physical interface, while **member** could identify a particular DS0 on that interface. If the **member** field is omitted, the gateway is expected to select an available facility in the specified **group**.

The **CarrierInfo** structure contains information about the Carrier Selection. The **carrierIdentificationCode** identifies the carrier (like carrier identification code in ISUP IAM message) chosen by the subscriber or determined by the routing applications, as a binary string of digits. The **carrierName** field is another means to identify the carrier as an ASCII string.

carrier – carrier identifier/selection code for call routing as determined by the routing applications or preferred by the subscriber.

The **ServiceControlDescriptor** structure contains service-specific data, or references to it, intended for user presentation or other service control communications as described, for example, in Annex K/H.323. The following options are possible:

- **url** – This selection contains a URL-referenced protocol or resource.
- **signal** – This selection contains a **SignalsDescriptor** as defined in ITU-T Rec. H.248, in binary format. The optional **streamID** and **notifyCompletion** elements shall be omitted from the **Signal** sequence in the **SignalsDescriptor**.
- **nonStandard** – This selection contains information not defined in this Recommendation (for example, proprietary data).
- **callCreditServiceControl** – This selection contains information related to controlling the duration of a call and advising the user of account balance information.

The **ServiceControlSession** structure contains a description of a service control session as described, for example, in Annex K/H.323. It contains the following fields:

- **sessionId** – An integer identifying this session that is unique for the client. Note that the identifiers received through different signalling paths (e.g., RAS and call signalling) are orthogonal and may overlap.
- **contents** – A **ServiceControl** structure with the relevant contents, or communication mechanism.

- **reason** – Indicates whether this is a new session (**open**) or a modification to an existing session (**refresh**), or that the session is being terminated by the provider (**close**) and existing resources such as a GUI, etc., should be closed.

The **RasUsageInfoTypes** structure lists types of usage information that may be reported by an endpoint to a gatekeeper. The endpoint uses this structure to indicate its capabilities with respect to collecting and reporting usage information, and the gatekeeper uses this structure to request usage information of particular types. The **nonStandardUsageTypes** field allows a vendor to refer to proprietary usage information types. The **startTime** and **endTime** fields refer to the times at which a call started and ended, respectively. The **terminationCause** parameter refers to the reason that the call ended.

The **RasUsageSpecification** structure is a template that allows a gatekeeper to request particular types of usage information at specific points in a call. The **when** field indicates the point or points in the call at which time the endpoint is requested to report the information; **start** refers to the start of the call, **end** refers to the end of the call, and **inIrr** refers to unsolicited IRR messages. The **callStartingPoint** field defines the point or points in the call that shall be considered the start of the call for the purposes of reporting usage information; a value of **connect** refers to transmission or reception of the Connect message, and a value of **alerting** refers to transmission or reception of the Alerting message. The **required** field indicates the types of usage information that the endpoint is required to report. A **RasUsageSpecification** structure in which nothing is selected in either the **when** or **required** fields indicates a request to disable the reporting of usage information.

The **RasUsageInformation** structure is a collection of usage data pertaining to a particular call. The **nonStandardUsageFields** field allows a vendor to list usage information of proprietary types. The **alertingTime** field indicates the time at which the Alerting message was sent or received. The **connectTime** field indicates the time at which the Connect message was sent or received. The **endTime** field indicates the time at which the Release Complete message was sent or received.

The **CallTerminationCause** structure indicates the reason for the end of a call. The **releaseCompleteReason** field indicates the **reason** that was specified in the Release Complete message. The **releaseCompleteCauseIE** field provides the Cause IE from the Release Complete message.

The **BandwidthDetails** structure defines additional bandwidth usage information that is not available in the **BandWidth** structure. The **sender** field is set to TRUE if the message is sent by the sender of the stream, or FALSE if sent by the receiver. The **multicast** field is set to TRUE if the stream is multicast, or FALSE otherwise. The **bandwidth** field indicates the bandwidth used for the stream in units of hundreds of bits per second. The **rtcpAddresses** field indicates the RTCP addresses used for the media stream.

The **CallCreditCapability** structure indicates certain capabilities of an endpoint related to billing for a call. By default, an endpoint is assumed not to have these optional capabilities. If a field in this structure is not included, this indicates that the status of the capability represented by that field has not changed since the last time it was reported. The **canDisplayAmountString** field indicates whether the endpoint can display a text string that contains the amount of currency in a user's account. The **canEnforceDurationLimit** field indicates whether an endpoint has the capability to disengage a call when a call duration limit indicated by the gatekeeper has elapsed.

The **CallCreditServiceControl** structure allows a gatekeeper to provide certain billing-related control and information to an endpoint. This structure provides the following fields:

- **amountString** – This field indicates the amount of money in a user's account, e.g., "$10.00". The string shall include the appropriate currency symbol. Note that standard abbreviations for currency types, such as "USD" for United States dollars, are defined by ISO 4217. The **amountString** field shall be encoded in Basic ISO/IEC 10646-1 (Unicode).

- **billingMode** – This field indicates the billing mode for this call. A mode of **debit** indicates that the call will result in charges against the amount of money available in a user's account. A mode of **credit** indicates that the call will result in charges to be paid at a later time. An endpoint could use this information, for example, to determine the type of announcement to play or display.
- **callDurationLimit** – This field indicates the remaining amount of time allowed for a particular call.
- **enforceCallDurationLimit** – This field indicates whether the endpoint is requested to disengage the call after the amount of time indicated by **callDurationLimit** has elapsed. If this field is not provided, the endpoint shall interpret this to indicate that the directive has not changed from its previous state.
- **callStartingPoint** – This field indicates the point in the call that timing is requested to begin if call duration enforcement is provided by the endpoint.

The **GenericData** structure consists of an **id** to identify the data, and the **parameters** field to convey the actual parameters.

The **GenericIdentifier** structure provides various ways to identify an object.

The **EnumeratedParameter** structure provides a generic parameter. It consists of an **id** to identify the parameter, and a **content** field to convey any associated data.

The **Content** structure supports a number of different data types, including **raw**, **text**, **unicode**, **bool**, **number8**, **number16**, **number32**, **id**, **alias**, **transport**, **compound** and **nested**. This allows for flexible definition of a generic parameter. The **raw** choice allows for a parameter or set of parameters whose actual data structure is defined elsewhere; for example, it could consist of PER-encoded ASN.1 or data in type-length-value form, or could be an encapsulated message of another signalling protocol.

The **FeatureSet** structure allows an entity to specify generic feature information. The entity specifies the set of features that it requires for successful completion of the call using the **neededFeatures** field, the set of features that it prefers but does not require using the **desiredFeatures** field, and the set of features that it supports in the **supportedFeatures** field. The **replacementFeatureSet** BOOLEAN is set to TRUE to indicate that this feature set replaces any previously sent feature set, or FALSE otherwise.

The **TransportChannelInfo** structure provides information about a media transport channel. The **sendAddress** field is the transport address of the sender, and the **recvAddress** is the transport address of the receiver.

The **RTPSession** structure provides a description of an RTP session. It has the following fields:

- **rtpAddress** – This field provides the send and receive addresses of the RTP stream.
- **rtcpAddress** – This field provides the send and receive addresses of RTCP messages.
- **cname** – This field provides the CNAME as specified in clause 6 and in Annex A.
- **ssrc** – This field is used to identify the source of an RTP stream, as described in clause 6 and in Annex A.
- **sessionId** – This field provides the identifier of this RTP session, as described in ITU-T Rec. H.245.
- **associatedSessionIds** – This field provides the identifiers of associated RTP sessions, as described in ITU-T Rec. H.245.
- **multicast** – This field indicates whether this is a multicast session.
- **bandwidth** – This field indicates the bandwidth used for the stream in units of hundreds of bits per second.

7.7 Required support of RAS messages

Table 22 shows the RAS messages that are supported by different endpoint types.

Table 22/H.225.0 – Status of RAS messages

RAS Message	Endpoint (Tx)	Endpoint (Rx)	Gatekeeper (Tx)	Gatekeeper (Rx)
GRQ	O			M
GCF		O	M	
GRJ		O	M	
RRQ	M			M
RCF		M	M	
RRJ		M	M	
URQ	O	M	O	M
UCF	M	O	M	O
URJ	O	O	M	O
ARQ	M			M
ACF		M	M	
ARJ		M	M	
BRQ	M	M	O	M
BCF	M (Note 1)	M	M	O
BRJ	M	M	M	O
IRQ		M	M	
IRR	M			M
IACK		O	CM	
INAK		O	CM	
DRQ	M	M	O	M
DCF	M	M	M	M
DRJ	M (Note 2)	M	M	M
LRQ	O		O	M
LCF		O	M	O
LRJ		O	M	O
NSM	O	O	O	O
XRS	M	M	M	M
RIP	CM	M	CM	M
RAI	O			M
RAC		O	M	
SCI	O	O	O	O
SCR	O	O	O	O

M: Mandatory, O: Optional, F: Forbidden, CM: Conditionally Mandatory, blank: "Not Applicable".

NOTE 1 – If a gatekeeper sends a BRQ requesting a lower rate, the endpoint shall reply with BCF if the lower rate is supported, otherwise with BRJ. If a gatekeeper sends a BRQ requesting a higher rate, the endpoint may reply with BCF or BRJ.

NOTE 2 – Terminal shall not send DRJ in response to a valid DRQ from its gatekeeper.

7.8 Terminal and gateway discovery messages

The GRQ message requests that any gatekeeper receiving it respond with a GCF granting it permission to register. The GRJ is a rejection of this request indicating that the requesting endpoint should seek another gatekeeper.

7.8.1 GatekeeperRequest (GRQ)

Note that one GRQ is sent per logical endpoint; thus, an MCU or a Gateway might send many.

The GRQ message includes the following:

requestSeqNum – This is a monotonically increasing number unique to the sender. It shall be returned by the receiver in any messages associated with this specific message.

protocolIdentifier – Identifies the H.225.0 vintage of the sending endpoint.

nonStandardData – Carries information not defined in this Recommendation (for example, proprietary data).

rasAddress – This is the transport address that this endpoint uses for registration and status messages. The Gatekeeper shall send RAS messages to this address and not to the address from which the message was sent, unless the **rasAddress** cannot be decoded.

endpointType – This specifies the type(s) of the endpoint that is registering (the MC bit shall not be set by itself).

gatekeeperIdentifier – String to identify the gatekeeper from which the terminal would like to receive permission to register. A missing or null string **gatekeeperIdentifier** indicates that the terminal is interested in any available gatekeeper.

callServices – Provides information on support of optional Q-series protocols to gatekeeper and called terminal.

endpointAlias – A list of alias addresses, by which other terminals may identify this terminal.

alternateEndpoints – A sequence of prioritized endpoint alternatives for **rasAddress**, **endpointType**, or **endpointAlias**.

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

authenticationCapability – This indicates the authentication mechanisms supported by the endpoint.

algorithmOIDs – Indicates the entire set of encryption algorithms supported by the endpoint.

integrity – Indicates to the recipient which integrity mechanism is to be applied on the RAS messages.

integrityCheckValue – Provides improved message integrity/message authentication of the RAS messages. The cryptographically-based integrity check value is computed by the sender applying a negotiated integrity algorithm and the secret key upon the entire message. Prior to **integrityCheckValue** computation, this field shall be ignored and shall be empty. After computation, the sender puts the computed integrity check value in the **integrityCheckValue** field and transmits the message.

supportsAltGK – Indicates whether the endpoint supports the alternate gatekeeper mechanism.

featureSet – This field specifies a set of generic features.

genericData – This field is a list of generic elements related to features that are defined outside of the base H.225.0 specification. These parameters may be used, for example, for tunnelling information transparently through RAS.

7.8.2 GatekeeperConfirm (GCF)

The GCF message includes the following:

requestSeqNum – This shall be the same value that was passed in the GRQ.

protocolIdentifier – Identifies the vintage of the accepting gatekeeper.

nonStandardData – Carries information not defined in this Recommendation (for example, proprietary data).

gatekeeperIdentifier – String to identify gatekeeper that is sending the GCF.

rasAddress – This is the transport address that the gatekeeper uses for registration and status messages.

alternateGatekeeper – Sequence of prioritized alternatives for **gatekeeperIdentifier** and **rasAddress**.

authenticationMode – This indicates the authentication mechanism to be used. The gatekeeper shall choose **authenticationMode** from **authenticationCapability** provided by the endpoint in GRQ.

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

algorithmOID – Indicates the encryption algorithm required by the gatekeeper.

integrity – Indicates to the recipient which integrity mechanism is to be applied on the RAS messages.

integrityCheckValue – Provides improved message integrity/message authentication of the RAS messages. The cryptographically-based integrity check value is computed by the sender applying a negotiated integrity algorithm and the secret key upon the entire message. Prior to **integrityCheckValue** computation, this field shall be ignored and shall be empty. After computation, the sender puts the computed integrity check value in the **integrityCheckValue** field and transmits the message.

featureSet – This field specifies a set of generic features.

genericData – This field is a list of generic elements related to features that are defined outside of the base H.225.0 specification. These parameters may be used, for example, for tunnelling information transparently through RAS.

7.8.3 GatekeeperReject (GRJ)

The GRJ message includes the following:

requestSeqNum – This shall be the same value that was passed in the GRQ.

protocolIdentifier – Identifies the vintage of the rejecting gatekeeper.

nonStandardData – Carries information not defined in this Recommendation (for example, proprietary data).

gatekeeperIdentifier – String to identify gatekeeper that is sending the GRJ.

rejectReason – Codes for why the GRQ was rejected by this gatekeeper. A reason of **genericDataReason** indicates that the request was rejected as a result of a generic element or feature; in this case, additional information may be specified in the **genericData** field.

altGKInfo – Optional information about alternative gatekeepers.

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

integrityCheckValue – Provides improved message integrity/message authentication of the RAS messages. The cryptographically-based integrity check value is computed by the sender applying a negotiated integrity algorithm and the secret key upon the entire message. Prior to **integrityCheckValue** computation, this field shall be ignored and shall be empty. After computation, the sender puts the computed integrity check value in the **integrityCheckValue** field and transmits the message.

featureSet – This field specifies a set of generic features.

genericData – This field is a list of generic elements related to features that are defined outside of the base H.225.0 specification. These parameters may be used, for example, for tunnelling information transparently through RAS.

7.9 Terminal and gateway registration messages

The RRQ is a request from a terminal to a gatekeeper to register. If the gatekeeper responds with a RCF, the terminal shall use the responding gatekeeper for future calls. If the gatekeeper responds with a RRJ, the terminal must seek another gatekeeper to register with.

7.9.1 RegistrationRequest (RRQ)

The RRQ message includes the following:

requestSeqNum – This is a monotonically increasing number unique to the sender. It shall be returned by the receiver in any response associated with this specific message.

protocolIdentifier – Identifies the H.225.0 vintage of the sending endpoint.

nonStandardData – Carries information not defined in this Recommendation (for example, proprietary data).

discoveryComplete – Set to TRUE if the requesting endpoint has preceded this message with the gatekeeper discovery procedure; set to FALSE if registering only. Note that registration may age, and the endpoint will get a failure on an RRQ or ARQ with a reason code of **discoveryRequired** or **notRegistered** respectively. This indicates that the endpoint should perform the discovery procedure (either dynamic or static) before issuing the RRQ with **discoveryComplete** set to TRUE.

callSignalAddress – This is the call signalling transport address for this endpoint. If multiple transports are supported, they shall be registered all at once.

rasAddress – This is the registration and status transport address for this endpoint. The Gatekeeper shall send RAS messages to this address and not to the address from which the message was sent, unless the **rasAddress** cannot be decoded.

terminalType – This specifies the type(s) of the endpoint that is(are) registering; note that the **mc** bit shall not be set by itself; either the **terminal**, **mcu**, **gateway**, or **gatekeeper** bit shall also be set. If **vendor** information is provided, this information shall be identical to that in **endpointVendor**. If the **terminalType** is **gateway** or **mcu**, then the optional **supportedPrefixes** value is a list of prefix addresses by which other endpoints may identify SCN protocols and data rates supported by this entity. This field may be used in addition to, or as an alternative to, the **terminalAlias** and **terminalAliasPattern** fields. All of the endpoint's supported prefixes shall be included in each RRQ unless the **additiveRegistration** option is specified, in which case the supported prefixes in an RRQ shall be added to the list of currently registered prefixes for the endpoint. With the additive RRQ, supported prefixes already registered to this endpoint shall be considered still registered. Note that prefixes are not part of a **PartyNumber** (E.164 or others). In order to register a **PartyNumber** (or a range or pattern of them), the endpoint shall use the **terminalAlias** and **terminalAliasPattern** fields as described below.

terminalAlias – This optional value is a list of alias addresses, by which other terminals may identify this terminal. This field may be used in addition to, or as an alternative to, the **terminalAliasPattern** and **supportedPrefixes** fields. If the **terminalAlias** is null, a **terminalAlias** address may be assigned by the gatekeeper, and included in the RCF. If an **email-ID** is available for the endpoint, it should be registered. Note that multiple alias addresses may refer to the same transport addresses. All of the endpoint's aliases that it desires to register shall be included in this list unless the **additiveRegistration** option is specified, in which case, the endpoint aliases in an RRQ shall be added to the list of aliases currently registered for the endpoint.

gatekeeperIdentifier – String to identify the gatekeeper that the terminal wishes to register with.

endpointVendor – Information about the endpoint vendor.

alternateEndpoints – A sequence of prioritized endpoint alternatives for **callSignalAddress**, **rasAddress**, **terminalType**, or **terminalAlias**.

timeToLive – Duration of the validity of the registration, in seconds. After this time, the gatekeeper may consider the registration stale.

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

integrityCheckValue – Provides improved message integrity/message authentication of the RAS messages. The cryptographically-based integrity check value is computed by the sender applying a negotiated integrity algorithm and the secret key upon the entire message. Prior to **integrityCheckValue** computation, this field shall be ignored and shall be empty. After computation, the sender puts the computed integrity check value in the **integrityCheckValue** field and transmits the message.

keepAlive – If set to TRUE, indicates that the endpoint has sent this RRQ as a "keep alive". An endpoint can send a lightweight RRQ consisting of only **rasAddress**, **keepAlive**, **endpointIdentifier**, **gatekeeperIdentifier**, **tokens**, and **timeToLive**. A gatekeeper in receipt of RRQ with a **keepAlive** field set to TRUE should ignore fields other than **endpointIdentifier**, **gatekeeperIdentifier**, **tokens**, and **timeToLive**. The **rasAddress** in a lightweight RRQ shall only be used by a gatekeeper as the destination for an RRJ when the endpoint is not registered.

endpointIdentifier – The **endpointIdentifier** provided by the gatekeeper during the original RCF.

willSupplyUUIEs – If set to TRUE, this indicates that the endpoint will supply H.225.0 call signalling message information in IRR messages if requested by the gatekeeper.

maintainConnection – If TRUE, this indicates that the sender of the message is capable of supporting a signalling connection when no calls are currently signalled over the connection.

alternateTransportAddresses – This field conveys call signalling addresses for transports other than TCP. Inclusion of an address indicates support for the corresponding transport.

additiveRegistration – If present, this field indicates that this message is an "additive" RRQ, meaning that the endpoint has sent this RRQ as an addition of information to an existing registration. An endpoint may send an additive RRQ consisting of only **callSignalAddress**, **rasAddress**, **terminalType**, **terminalAlias**, **terminalAliasPattern**, **alternateEndpoints**, **endpointIdentifier**, **gatekeeperIdentifier**, and **tokens**. A gatekeeper in receipt of an RRQ with the **additiveRegistration** field present shall ignore fields other than these. The **rasAddress** in an additive RRQ shall be used by a gatekeeper as the destination for the subsequent RRJ if the endpoint is not registered, or if the **terminalAlias** and/or **terminalAliasPattern** conflicts with the gatekeeper's registration policy.

terminalAliasPattern – This optional value is a list of address patterns specifying aliases and addresses by which other endpoints may identify this endpoint. This field may be used in addition to, or as an alternative to, the **terminalAlias** and **supportedPrefixes** fields. All of the endpoint's aliases and addresses shall be included in each RRQ unless the **additiveRegistration** option is TRUE, in which case the endpoint aliases and addresses in the RRQ shall be added to the list of aliases currently registered for the endpoint.

supportsAltGK – Indicates whether the endpoint supports the alternate gatekeeper mechanism.

usageReportingCapability – This field may be included by the endpoint to advertise its ability to collect and report various types of usage information.

multipleCalls – If TRUE, this field indicates that the sender of the message is capable of signalling multiple calls over a single call signalling connection.

supportedH248Packages – This field indicates a list of H.248 packages supported by this endpoint.

callCreditCapability – This field describes certain billing-related capabilities of this endpoint.

capacityReportingCapability – This field describes the endpoint's ability to report call capacity information.

capacity – This field indicates the endpoint's maximum and current call capacity. When sending this field, the endpoint shall include the **maximumCallCapacity** and **currentCallCapacity** elements.

featureSet – This field specifies a set of generic features.

genericData – This field is a list of generic elements related to features that are defined outside of the base H.225.0 specification. These parameters may be used, for example, for tunnelling information transparently through RAS.

restart – If set, this field indicates that this is the first RRQ sent by the endpoint after its reboot, or after an abnormal event that resulted in loss of its calls. This allows the gatekeeper to perform any cleanup or other functions as necessary.

supportsACFSequences – If set, this field indicates that the endpoint is able to receive and process a sequence of ACF messages in response to a single ARQ message.

7.9.2 RegistrationConfirm (RCF)

The RCF message includes the following:

requestSeqNum – This shall be the same value that was passed in the RRQ.

protocolIdentifier – Identifies the vintage of the accepting gatekeeper.

nonStandardData – Carries information not defined in this Recommendation (for example, proprietary data).

callSignalAddress – This is an array of transport addresses for H.225.0 call signalling messages; one for each transport that the gatekeeper will respond to. This address includes the TSAP identifier.

terminalAlias – This optional value is a list of alias addresses, by which other terminals may identify this terminal. This field may be used in addition to, or as an alternative to, the **terminalAliasPattern** and **supportedPrefixes** fields. It specifies the alias addresses that have been accepted from those proposed in the associated RRQ message. If none were proposed in the RRQ, this list gives aliases assigned by the gatekeeper. If this field is not included, and alias addresses were proposed in the RRQ, then the gatekeeper has accepted all of the proposed alias addresses. If this field is included and specifies a subset of the alias addresses proposed in the RRQ, then the Gatekeper has accepted only those addresses.

gatekeeperIdentifier – String to identify the gatekeeper that has accepted the terminal's registration.

endpointIdentifier – A gatekeeper assigned terminal identity string; shall be echoed in subsequent RAS messages.

alternateGatekeeper – Sequence of prioritized alternatives for **gatekeeperIdentifer** and **rasAddress**.

timeToLive – Duration of the validity of the registration, in seconds. After this time the gatekeeper may consider the registration stale.

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

integrityCheckValue – Provides improved message integrity/message authentication of the RAS messages. The cryptographically-based integrity check value is computed by the sender applying a negotiated integrity algorithm and the secret key upon the entire message. Prior to **integrityCheckValue** computation, this field shall be ignored and shall be empty. After computation, the sender puts the computed integrity check value in the **integrityCheckValue** field and transmits the message.

willRespondToIRR – True if the gatekeeper will send an IACK or INAK message in response to an unsolicited IRR message with its **needsResponse** field set to TRUE.

preGrantedARQ – Indicates events for which the gatekeeper has pre-granted admission. This allows for faster call setup times in environments where admission is guaranteed through means other than the ARQ/ACF exchange. Note that even if these fields are set to TRUE, an endpoint can still send an ARQ to the gatekeeper for reasons such as address translation, or the endpoint does not support this modified signalling mode. If the **preGrantedARQ** sequence is not present, then ARQ signalling shall be used in all cases. The fields are:

- **makeCall** – If the **makeCall** flag is TRUE, then the gatekeeper has pre-granted permission to the endpoint to initiate calls without first sending an ARQ. If the **makeCall** flag is FALSE, the endpoint shall always send ARQ to get permission to make a call.
- **useGKCallSignalAddressToMakeCall** – If the **makeCall** and **useGKCallSignalAddressToMakeCall** flags are both set to TRUE, then if the endpoint does not send an ARQ to the gatekeeper to make a call, the endpoint shall send all H.225 call signalling to the gatekeeper call signalling channel.
- **answerCall** – If the **answerCall** flag is TRUE, then the gatekeeper has pre-granted permission to the endpoint to answer calls without first sending an ARQ. If the **answerCall** flag is FALSE, the endpoint shall always send ARQ to get permission to answer a call.
- **useGKCallSignalAddressToAnswer** – If the **answerCall** and **useGKCallSignalAddressToAnswer** flags are both set to TRUE, then when an endpoint does not send an ARQ to the gatekeeper to answer a call, the endpoint shall ensure that all H.225.0 call signalling comes from the gatekeeper. If an endpoint has been instructed to use the gatekeeper when answering, but it does not know whether an incoming call has come from the gatekeeper (which may involve looking at the transport address), the endpoint shall issue ARQ irrespective of the state of the **useGKCallSignalAddressToAnswer** flag.
- **irrFrequencyInCall** – This indicates the frequency, in seconds, of IRR messages sent to the gatekeeper when the endpoint is in one or more calls. If it is not present, the gatekeeper does not want unsolicited IRR messages. When the endpoint is sending these IRR messages, the call reference value shall be made unique for the terminal, as it would have been generated in an Admission Request. However, this is not a "normal" CRV, and cannot be reused for further communication (DRQ, IRQ or BRQ). The call identifier shall be the same as used in the call signalling channel messages for the related call.

- **totalBandwidthRestriction** – This field limits the total use of bandwidth for the endpoint when in calls. If it is not present, there is no constant bandwidth restriction.
- **alternateTransportAddresses** – This field conveys call signalling addresses for transports other than TCP. Inclusion of an address indicates support for the corresponding transport.
- **useSpecifiedTransport** – This field allows the gatekeeper to instruct the endpoint as to which signalling transport protocol to use for making calls. If this field is included and the specified transport is not **tcp**, then the **alternateTransportAddresses** shall also be included in this message.

maintainConnection – If TRUE, this indicates that the gatekeeper (in the case of gatekeeper routing) is capable of supporting a signalling connection when no calls are currently signalled over the connection.

serviceControl – Contains service specific data or addressing information that the endpoint may use for non-call related service control communication with the network as described, for example, in Annex K/H.323.

supportsAdditiveRegistration – If present, this field indicates that the gatekeeper supports additive registration capabilities. If not present, the gatekeeper does not support additive registration.

terminalAliasPattern – This optional value is a list of address patterns specifying aliases and addresses by which other endpoints may identify this endpoint. This field may be used in addition to, or as an alternative to, the **terminalAlias** and **supportedPrefixes** fields. It specifies the aliases and addresses that have been accepted from those proposed in the associated RRQ message. If none were proposed in the RRQ, this list gives aliases and addresses assigned by the gatekeeper. If this field is not included and address patterns were proposed in the RRQ, then the gatekeeper has accepted all of the proposed patterns. If this field is included and specifies a subset of the address patterns proposed in the RRQ, then the gatekeeper has accepted only those patterns.

supportedPrefixes – This optional value is a list of prefixes by which other endpoints may identify this endpoint. This field may be used in addition to or as an alternative to the **terminalAlias** and **terminalAliasPattern** fields. It specifies the address prefixes that have been accepted from those proposed in the associated RRQ message. If none were proposed in the RRQ, this list gives prefixes assigned by the gatekeeper. If this field is not included and address prefixes were proposed in the RRQ, then the gatekeeper has accepted all of the proposed prefixes. If this field is included and specifies a subset of the address prefixes proposed in the RRQ, then the gatekeeper has accepted only those prefixes.

usageSpec – This field may be included by the gatekeeper to request that the endpoint collect and report the indicated call usage information at the specified points in time.

featureServerAlias – This field is reserved for future use by the ITU-T for a stimulus-based protocol.

capacityReportingSpec – This field indicates the type of call capacity information that an endpoint is requested to report.

featureSet – This field specifies a set of generic features.

genericData – This field is a list of generic elements related to features that are defined outside of the base H.225.0 specification. These parameters may be used, for example, for tunnelling information transparently through RAS.

7.9.3 RegistrationReject (RRJ)

The RRJ message includes the following:

requestSeqNum – This shall be the same value that was passed in the RRQ.

protocolIdentifier – Identifies the vintage of the rejecting gatekeeper.

nonStandardData – Carries information not defined in this Recommendation (for example, proprietary data).

rejectReason – The reason for the rejection of the registration. This field may contain an **invalidTerminalAliases** value, in which case it contains a list of aliases, addresses and supported prefixes that were determined to be invalid in the associated RRQ message. In any event, all of the aliases, addresses and supported prefixes from the associated RRQ are rejected along with those specified in the **invalidTerminalAliases** field. A reason of **genericDataReason** indicates that the request was rejected as a result of a generic element or feature; in this case, additional information may be specified in the **genericData** field.

gatekeeperIdentifier – String to identify the gatekeeper that has rejected the terminal's registration.

altGKInfo – Optional information about alternative gatekeepers.

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

integrityCheckValue – Provides improved message integrity/message authentication of the RAS messages. The cryptographically-based integrity check value is computed by the sender applying a negotiated integrity algorithm and the secret key upon the entire message. Prior to **integrityCheckValue** computation, this field shall be ignored and shall be empty. After computation, the sender puts the computed integrity check value in the **integrityCheckValue** field and transmits the message.

featureSet – This field specifies a set of generic features.

genericData – This field is a list of generic elements related to features that are defined outside of the base H.225.0 specification. These parameters may be used, for example, for tunnelling information transparently through RAS.

7.10 Terminal/gatekeeper unregistration messages

7.10.1 UnregistrationRequest (URQ)

The URQ requests that the association between a terminal and a gatekeeper be broken. Note that unregister is bidirectional, i.e., a gatekeeper can request a terminal to consider itself unregistered, and a terminal can inform a gatekeeper that it is revoking a previous registration.

The URQ message includes the following:

requestSeqNum – This is a monotonically increasing number unique to the sender. It shall be returned by the receiver in any response associated with this specific message.

callSignalAddress – This is one or more of the transport call signalling addresses for this endpoint which are to be unregistered.

endpointAlias – This optional value is a list of alias addresses, by which other terminals may identify this terminal. This field may be used in addition to or as an alternative to the **endpointAliasPattern** and **supportedPrefixes** fields. If this field, the **endpointAliasPattern** field, and the **supportedPrefixes** field are not present, all aliases are unregistered in a single message. The **dialledDigits** value, if assigned, is required. Only values listed here are unregistered; this allows, for example, an **h323-ID** to be unregistered while leaving the **dialledDigits** value registered.

nonStandardData – Carries information not defined in this Recommendation (for example, proprietary data).

endpointIdentifier – Confirmation of identity; not sent by the gatekeeper.

alternateEndpoints – A sequence of prioritized endpoint alternatives for **callSignalAddress** or **endpointAlias**.

gatekeeperIdentifier – A **gatekeeperIdentifier** which the endpoint received in the **alternateGatekeeper** list in an RCF from the gatekeeper when it registered or in a previous URJ message.

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

integrityCheckValue – Provides improved message integrity/message authentication of the RAS messages. The cryptographically-based integrity check value is computed by the sender applying a negotiated integrity algorithm and the secret key upon the entire message. Prior to **integrityCheckValue** computation, this field shall be ignored and shall be empty. After computation, the sender puts the computed integrity check value in the **integrityCheckValue** field and transmits the message.

reason – Used when the gatekeeper sends the URQ to indicate why the gatekeeper considers the endpoint unregistered. A **reason** of **maintenance** indicates that the gatekeeper or endpoint is being taken down for maintenance.

endpointAliasPattern – This optional value is a list of address patterns specifying aliases and addresses by which other endpoints may identify this endpoint. This field may be used in addition to or as an alternative to the **endpointAlias** and **supportedPrefixes** fields. If this field, the **endpointAlias** field and the **supportedPrefixes** field are not present, all aliases and addresses are unregistered in a single message. Otherwise, only values listed here are unregistered.

supportedPrefixes – This optional value is a list of prefixes by which other endpoints may identify this endpoint. This field may be used in addition to or as an alternative to the **terminalAlias** and **terminalAliasPattern** fields. If this field, the **endpointAlias** field and the **endpointAliasPattern** field are not present, all aliases and addresses are unregistered in a single message. Otherwise, only values listed here are unregistered.

alternateGatekeeper – Sequence of prioritized alternatives for **gatekeeperIdentifier** and **rasAddress**.

genericData – This field is a list of generic elements related to features that are defined outside of the base H.225.0 specification. These parameters may be used, for example, for tunnelling information transparently through RAS.

7.10.2 UnregistrationConfirm (UCF)

The UCF message includes the following:

requestSeqNum – This shall be the same value that was passed in the URQ.

nonStandardData – Carries information not defined in this Recommendation (for example, proprietary data).

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

integrityCheckValue – Provides improved message integrity/message authentication of the RAS messages. The cryptographically-based integrity check value is computed by the sender applying a negotiated integrity algorithm and the secret key upon the entire message. Prior to **integrityCheckValue** computation, this field shall be ignored and shall be empty. After computation, the sender puts the computed integrity check value in the **integrityCheckValue** field and transmits the message.

genericData – This field is a list of generic elements related to features that are defined outside of the base H.225.0 specification. These parameters may be used, for example, for tunnelling information transparently through RAS.

7.10.3 UnregistrationReject (URJ)

The URJ message includes the following:

requestSeqNum – This shall be the same value that was passed in the URQ.

rejectReason – The reason for the rejection of the unregistration.

nonStandardData – Carries information not defined in this Recommendation (for example, proprietary data).

altGKInfo – Optional information about alternative gatekeepers.

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

integrityCheckValue – Provides improved message integrity/message authentication of the RAS messages. The cryptographically-based integrity check value is computed by the sender applying a negotiated integrity algorithm and the secret key upon the entire message. Prior to **integrityCheckValue** computation, this field shall be ignored and shall be empty. After computation, the sender puts the computed integrity check value in the **integrityCheckValue** field and transmits the message.

genericData – This field is a list of generic elements related to features that are defined outside of the base H.225.0 specification. These parameters may be used, for example, for tunnelling information transparently through RAS.

7.11 Terminal to gatekeeper admission messages

The ARQ message requests that an endpoint be allowed access to the packet-based network by the gatekeeper, which either grants the request with an ACF or denies it with an ARJ.

7.11.1 AdmissionRequest (ARQ)

The ARQ message includes the following:

requestSeqNum – This is a monotonically increasing number unique to the sender. It shall be returned by the receiver in any messages associated with this specific message.

callType – Using this value, the gatekeeper can attempt to determine "real" bandwidth usage. The default value is **pointToPoint** for all calls. It should be recognized that the call type may change dynamically during the call and that the final call type may not be known when the ARQ is sent.

callModel – If **direct**, the endpoint is requesting the direct terminal to terminal call model. If **gatekeeperRouted**, the endpoint is requesting the gatekeeper mediated model. The gatekeeper is not required to comply with this request.

endpointIdentifier – This is an endpoint identifier that was assigned to the terminal by RCF.

destinationInfo – Sequence of alias addresses for the destination, such as **dialledDigits**, **PartyNumber** (**e164Number** or **privateNumber**), or **h323-ID**s. When sending the ARQ to answer a call, **destinationInfo** indicates the destination of the call (the answering endpoint). If at least one alias is registered with a gatekeeper and no two aliases in the ARQ are registered to distinct people, the gatekeeper shall recognize the ARQ as referring to the registered identity. In the case of conflicting aliases the admission request should be rejected with cause **AliasesInconsistent**. If the gatekeeper does not provide this validation, it shall consider the first registered address to be the destination.

destCallSignalAddress – Transport address used at the destination for call signalling.

destExtraCallInfo – Contains external addresses for multiple calls.

srcInfo – Sequence of alias addresses for the source endpoint, such as **dialledDigits**, **PartyNumber** (**e164Number** or **privateNumber**) or **h323-ID**s. When sending the ARQ to answer a call, **srcInfo** indicates the originator of the call.

srcCallSignalAddress – Transport address used at the source for call signalling.

bandWidth – The bidirectional bandwidth requested for the call, in units of 100 bits per second. For example, a 128 kbit/s call would be signalled as a request for 256 kbit/s. The value refers only to the audio and video bit rate excluding headers and overhead.

callReferenceValue – The CRV from H.225.0 call signalling messages for this call; only local validity. This is used by a gatekeeper to associate the ARQ with a particular call.

nonStandardData – Carries information not defined in this Recommendation (for example, proprietary data).

callServices – Provides information on support of optional Q-series protocols to gatekeeper and called terminal.

conferenceID – Unique conference identifier.

activeMC – If TRUE, the calling party has an active MC; otherwise, FALSE.

answerCall – Used to indicate to a gatekeeper that a call is incoming.

canMapAlias – If set to TRUE, indicates that if the resulting ACF contains **destinationInfo**, **destExtraCallInfo** and/or **remoteExtensionAddress** fields, the endpoint shall copy this information to the **destinationAddress**, **destExtraCallInfo** and **remoteExtensionAddress** fields of the Setup message respectively, or into the Called Party Number IE if appropriate. If the endpoint is a gateway used to exit the H.323 network, the gateway will convert the destination information into the appropriate signalling format used outside of the H.323 network (for example, DTMF). If the GK would replace addressing information from the ARQ and **canMapAlias** is FALSE, then the gatekeeper should reject the ARQ. Systems compliant with H.225.0 version 4 and higher shall set this field to TRUE.

callIdentifier – A globally unique call identifier set by the originating endpoint which can be used to associate RAS signalling with the modified Q.931 signalling used in this Recommendation.

srcAlternatives – A sequence of prioritized source endpoint alternatives for **srcInfo**, **srcCallSignalAddress**, or **rasAddress**.

destAlternatives – A sequence of prioritized destination endpoint alternatives for **destinationInfo** or **destCallSignalAddress**.

gatekeeperIdentifier – A **gatekeeperIdentifier** which the endpoint received in the **alternateGatekeeper** list in an RCF from the gatekeeper when it registered or in a previous ARJ message.

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

integrityCheckValue – Provides improved message integrity/message authentication of the RAS messages. The cryptographically-based integrity check value is computed by the sender applying a negotiated integrity algorithm and the secret key upon the entire message. Prior to **integrityCheckValue** computation, this field shall be ignored and shall be empty. After computation, the sender puts the computed integrity check value in the **integrityCheckValue** field and transmits the message.

transportQOS – An endpoint may use this to indicate its capability to reserve transport resources. The TransportQOS structure includes the following:

- **endpointControlled** – The endpoint will apply its own reservation mechanism.
- **gatekeeperControlled** – The gatekeeper will perform resource reservation on behalf of the endpoint.
- **noControl** – No resource reservation is needed.

willSupplyUUIEs – If set to TRUE, this indicates that the endpoint will supply H.225.0 call sginalling message information in IRR messages if requested by the gatekeeper.

callLinkage – The contents of this field are typically controlled by a call linkage service. For the procedures and semantics of this field, refer to clause 10/H.323.

gatewayDataRate – The requested data rate for the SCN side of a call through a gateway. This data rate, if present, shall be equal to the data rate specified in the Bearer capability IE of the Setup message. A gatekeeper might use this field in selecting a gateway to handle the call.

capacity – This field indicates the sending endpoint's available call capacity at this point in time, assuming that the gatekeeper confirms the ARQ by sending an ACF. When sending this field, the endpoint shall include the **currentCallCapacity** element.

circuitInfo – This field provides information about the SCN circuit or circuits used for this call.

desiredProtocols – Identifies the type of protocols, in order of preference, the originating endpoint desires for its call (e.g., voice or fax). A resolving entity may use this field to locate an endpoint that also supports the protocol, giving consideration to the order of preference.

desiredTunnelledProtocol – This field identifies a protocol that is requested to be tunnelled.

featureSet – This field specifies a set of generic features that relate to this call.

genericData – This field is a list of generic elements related to features that are defined outside of the base H.225.0 specification. These parameters may be used, for example, for tunnelling information transparently through RAS.

canMapSrcAlias – If set to TRUE, indicates that if the resulting ACF contains **modifiedSrcInfo**, the endpoint shall copy this information to the **sourceInfo** field of the Setup message and/or to the Calling Party Number IE, if appropriate. If the gatekeeper would replace addressing information from the ARQ and **canMapSrcAlias** is FALSE, then the gatekeeper should reject the ARQ.

NOTE – Both **destinationInfo** and **destCallSignalAddress** are optional, but at least one shall be present unless the endpoint is answering a call. There is no absolute rule which is preferred as this may be site-specific, but the address should be provided if available. It is cautioned that the best results will be obtained by considering the nature of the transport protocols in use.

7.11.2 AdmissionConfirm (ACF)

The ACF message includes the following:

requestSeqNum – This shall be the same value that was passed in the ARQ.

bandWidth – The allowed maximum bandwidth for the call; may be less than that requested.

callModel – Tells terminal whether call signalling sent on **destCallSignalAddress** goes to a gatekeeper or to a terminal. A value of **gatekeeperRouted** indicates that call signalling is being passed via the gatekeeper, while **direct** indicates that the endpoint-to-endpoint call mode is in use.

destCallSignalAddress – The transport address to which to send H.225.0 call signalling, but may be an endpoint or gatekeeper address depending on the call model in use.

irrFrequency – The frequency, in seconds, that the endpoint shall send IRRs to the gatekeeper while on a call, including while on hold. If not present, the endpoint does not send IRRs while active on a call, and it is expected that the gatekeeper will poll the endpoint.

nonStandardData – Carries information not defined in this Recommendation (for example, proprietary data).

destinationInfo – The address of the initial channel, used when calling through a gateway.

destExtraCallInfo – Needed to make possible additional channel calls, i.e., for a 2 × 64 kbit/s call on the SCN side. Shall only contain **dialledDigits** or **PartyNumber** addresses and shall not contain the number of the initial channel.

destinationType – This specifies the type of the destination endpoint.

remoteExtensionAddress – Contains the alias address of a called endpoint in cases where this information is needed to traverse multiple gateways.

alternateEndpoints – A sequence of prioritized endpoint alternatives for **destCallSignalAddress** or **destinationInfo**.

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

integrityCheckValue – Provides improved message integrity/message authentication of the RAS messages. The cryptographically-based integrity check value is computed by the sender applying a negotiated integrity algorithm and the secret key upon the entire message. Prior to **integrityCheckValue** computation, this field shall be ignored and shall be empty. After computation, the sender puts the computed integrity check value in the **integrityCheckValue** field and transmits the message.

transportQOS – The gatekeeper may indicate to the endpoint where the responsibility lies for resource reservation. If the gatekeeper received a **TransportQOS** in ARQ, then it should include **transportQOS** (possibly modified according to gatekeeper implementation) in ACF.

willRespondToIRR – TRUE if the gatekeeper will send an IACK or INAK message in response to an unsolicited IRR message when the IRR's **needsResponse** field set to TRUE.

uuiesRequested – The gatekeeper may request the endpoint to notify the gatekeeper of H.225.0 call signalling messages that the endpoint sends or receives if the endpoint indicated this capability in the ARQ by setting **willSupplyUUIEs** to TRUE. **uuiesRequested** indicates the set of H.225.0 call signalling messages of which the endpoint shall notify the gatekeeper.

language – Indicates the language(s) in which the user would prefer to receive announcements and prompts. The field contains one or more RFC 1766 compliant language tags.

alternateTransportAddresses – This field conveys call signalling addresses for transports other than TCP. Inclusion of an address indicates support for the corresponding transport.

useSpecifiedTransport – This field allows the gatekeeper to instruct the endpoint as to which signalling transport protocol to use for making the call. If this field is included and the specified transport is not **tcp**, then the **alternateTransportAddresses** shall also be included in this message.

circuitInfo – This field provides information about the SCN circuit or circuits used for this call. For example, it allows a gatekeeper to instruct an egress gateway to select particular SCN facilities to be used for the call.

usageSpec – This field may be included by the gatekeeper to request that the endpoint collect and report the indicated call usage information at the specified points in time in this call.

supportedProtocols – This field indicates the protocols supported by the destination endpoint.

serviceControl – This field contains service-specific data, or references to it, that could be used by an endpoint (e.g., a message to be played to the caller) as described, for example, in Annex K/H.323.

multipleCalls – If TRUE, this field indicates that the destination endpoint is capable of signalling multiple calls over a single call signalling connection. If FALSE, the destination endpoint does not have this capability. If this field is not present, the gatekeeper does not know whether the remote endpoint has this capability.

featureSet – This field specifies a set of generic features that relate to this call.

genericData – This field is a list of generic elements related to features that are defined outside of the base H.225.0 specification. These parameters may be used, for example, for tunnelling information transparently through RAS.

modifiedSrcInfo – Alias address that should be used for the source endpoint, such as **dialledDigits**, **PartyNumber** (**e164Number** or **privateNumber**) or **h323-ID**s. This field should be used when the calling endpoint's alias address is translated/modified when attempting to route the call to the primary destination, or to any of the alternate endpoints. These addresses should be used by the endpoint for this call only.

7.11.3 AdmissionReject (ARJ)

The ARJ message includes the following:

requestSeqNum – This shall be the same value that was passed in the ARQ.

rejectReason – This is the reason the admission request was denied. Note that the **rejectReason** of **routeCallToSCN** is an appropriate choice only when the ARJ is directed to an ingress gateway (the ARQ was sent by a gateway and the **answerCall** BOOLEAN in the ARQ is FALSE). If **rejectReason** is **routeCallToSCN**, the **rejectReason** for this choice also includes a telephone number, or list of telephone numbers, to which the gateway can redirect the call in the SCN, if the gateway supports such a procedure. If **rejectReason** is **exceedsCallCapacity**, the gatekeeper has determined that the destination does not have the capacity to accept this call at this point in time. A **rejectReason** of **collectDestination** indicates that the gatekeeper is requesting that the gateway collect the final destination address, and that the **serviceControl** field of the ARJ indicates the prompt to be presented to the user. A **rejectReason** of **collectPIN** indicates that the gatekeeper is requesting that the gateway collect a personal identification number or authorization code, and that the **serviceControl** field of the ARJ indicates the prompt to be presented to the user. A reason of **genericDataReason** indicates that the request was rejected as a result of a generic element or feature; in this case, additional information may be specified in the **genericData** field. The endpoint should reregister with the gatekeeper if it receives an error of **invalidEndpointIdentifier**.

nonStandardData – Carries information not defined in this Recommendation (for example, proprietary data).

altGKInfo – Optional information about alternative gatekeepers.

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

callSignalAddress – This is the gatekeeper's call signalling address returned when the reject reason is **routeCallToGatekeeper**.

integrityCheckValue – Provides improved message integrity/message authentication of the RAS messages. The cryptographically-based integrity check value is computed by the sender applying a negotiated integrity algorithm and the secret key upon the entire message. Prior to **integrityCheckValue** computation, this field shall be ignored and shall be empty. After computation, the sender puts the computed integrity check value in the **integrityCheckValue** field and transmits the message.

serviceControl – This field contains service-specific data, or references to it, that could be used by an endpoint (e.g., for displaying the reason a call failed) as described, for example, in Annex K/H.323.

featureSet – This field specifies a set of generic features that relate to this call.

genericData – This field is a list of generic elements related to features that are defined outside of the base H.225.0 specification. These parameters may be used, for example, for tunnelling information transparently through RAS.

7.12 Terminal to gatekeeper requests for changes in bandwidth

The BRQ message requests that an endpoint be granted a changed packet-based network bandwidth allocation by the gatekeeper, which either grants the request with a BCF or denies it with a BRJ.

The gatekeeper may request that an endpoint raise or lower the bandwidth in use with a BRQ. If the request is to raise the rate, the endpoint may reply with either BRJ or BCF. If the request is for a lower rate, the endpoint shall reply with a BCF if the lower rate is supported, otherwise with BRJ.

7.12.1 BandwidthRequest (BRQ)

The BRQ message includes the following:

requestSeqNum – This is a monotonically increasing number unique to the sender. It shall be returned by the receiver in any messages associated with this specific message.

endpointIdentifier – This is an endpoint identifier that was assigned to the terminal by RCF.

conferenceID – ID of the call that is to have the bandwidth changed.

callReferenceValue – The CRV from H.225.0 call signalling messages for this call; only local validity. This is used by a gatekeeper to associate the BRQ with a particular call.

callType – Using this value, the gatekeeper can attempt to determine "real" bandwidth usage.

bandWidth – The new bidirectional bandwidth requested for the call, in units of 100 bits per second. This is an absolute value that includes only audio and video bitstreams not counting headers and overhead. Unique multicast streams shall only add to the total bandwidth usage one time, even if there are multiple recipients of the media stream.

nonStandardData – Carries information not defined in this Recommendation (for example, proprietary data).

callIdentifier – A globally unique call identifier set by the originating endpoint which can be used to associate RAS signalling with the modified Q.931 signalling used in this Recommendation.

gatekeeperIdentifier – A **gatekeeperIdentifier** which the endpoint received in the **alternateGatekeeper** list in an RCF from the gatekeeper when it registered or in a previous BRJ message.

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

integrityCheckValue – Provides improved message integrity/message authentication of the RAS messages. The cryptographically-based integrity check value is computed by the sender applying a negotiated integrity algorithm and the secret key upon the entire message. Prior to **integrityCheckValue** computation, this field shall be ignored and shall be empty. After computation, the sender puts the computed integrity check value in the **integrityCheckValue** field and transmits the message.

answeredCall – Set to TRUE to indicate that this party was the original destination (this party answered the call).

callLinkage – The contents of this field are typically controlled by a call linkage service. For the procedures and semantics of this field, refer to clause 10/H.323.

capacity – This field indicates the sending endpoint's available call capacity at this point in time, assuming that the gatekeeper confirms the BRQ by sending a BCF. When sending this field, the endpoint shall include the **currentCallCapacity** element.

usageInformation – This field allows the endpoint to report usage information for this call. A gatekeeper shall not include this field when sending a BRQ.

bandwidthDetails – Provides bandwidth information for each media stream that the endpoint is currently transmitting or receiving in the same units as the **bandWidth** field. Each multicast stream shall be reported only one time, even if there are multiple recipients of the media stream.

genericData – This field is a list of generic elements related to features that are defined outside of the base H.225.0 specification. These parameters may be used, for example, for tunnelling information transparently through RAS.

7.12.2 BandwidthConfirm (BCF)

The BCF message includes the following:

requestSeqNum – This shall be the same value that was passed in the BRQ.

bandWidth – The maximum allowed at this time in increments of 100 bits.

nonStandardData – Carries information not defined in this Recommendation (for example, proprietary data).

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

integrityCheckValue – Provides improved message integrity/message authentication of the RAS messages. The cryptographically-based integrity check value is computed by the sender applying a negotiated integrity algorithm and the secret key upon the entire message. Prior to **integrityCheckValue** computation, this field shall be ignored and shall be empty. After computation, the sender puts the computed integrity check value in the **integrityCheckValue** field and transmits the message.

capacity – This field indicates the sending endpoint's available call capacity at this point in time. When sending this field, the endpoint shall include the **currentCallCapacity** element. This field is not included when the BCF is sent by a gatekeeper.

genericData – This field is a list of generic elements related to features that are defined outside of the base H.225.0 specification. These parameters may be used, for example, for tunnelling information transparently through RAS.

7.12.3 BandwidthReject (BRJ)

The BRJ message includes the following:

requestSeqNum – This shall be the same value that was passed in the BRQ.

rejectReason – The reason the change was rejected by the gatekeeper.

allowedBandWidth – The maximum allowed at this time in increments of 100 bits including the current allocation.

nonStandardData – Carries information not defined in this Recommendation (for example, proprietary data).

altGKInfo – Optional information about alternative gatekeepers.

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

integrityCheckValue – Provides improved message integrity/message authentication of the RAS messages. The cryptographically-based integrity check value is computed by the sender applying a negotiated integrity algorithm and the secret key upon the entire message. Prior to **integrityCheckValue** computation, this field shall be ignored and shall be empty. After computation, the sender puts the computed integrity check value in the **integrityCheckValue** field and transmits the message.

genericData – This field is a list of generic elements related to features that are defined outside of the base H.225.0 specification. These parameters may be used, for example, for tunnelling information transparently through RAS.

7.13 Location request messages

The LRQ requests that a gatekeeper provide address translation. The gatekeeper responds with an LCF containing the transport address of the destination, or rejects the request with LRJ.

7.13.1 LocationRequest (LRQ)

The LRQ message includes the following:

requestSeqNum – This is a monotonically increasing number unique to the sender. It shall be returned by the receiver in any messages associated with this specific message.

endpointIdentifier – This is an endpoint identifier that was assigned to the terminal by RCF.

destinationInfo – Sequence of alias addresses for the destination, such as as **dialledDigits**, **partyNumber** (**e164Number** or **privateNumber**), or **h323-ID**s. If at least one alias is registered with a gatekeeper and no two aliases in the LRQ are registered to distinct people, the gatekeeper shall recognize the LRQ as referring to the registered identity. In the case of conflicting aliases, the location request should be rejected with cause **AliasesInconsistent**. If the gatekeeper does not provide this validation, it shall consider the first registered address to be the destination.

nonStandardData – Carries information not defined in this Recommendation (for example, proprietary data).

replyAddress – Transport address to which to send the LCF/LRJ.

sourceInfo – Indicates the sender of the LRQ. The gatekeeper can use this information to decide how to respond to the LRQ.

canMapAlias – If set to TRUE, indicates that if the resulting LCF contains **destinationInfo**, **destExtraCallInfo** and/or **remoteExtensionAddress** fields, the endpoint can copy this information to the **destinationAddress**, **destExtraCallInfo** and **remoteExtensionAddress** fields of the Setup message respectively. If the GK would replace addressing information from the LRQ and **canMapAlias** is FALSE, then the gatekeeper should reject the LRQ. Systems compliant with H.225.0 version 4 and higher shall set this field to TRUE.

gatekeeperIdentifier – A **gatekeeperIdentifier** which the endpoint received in the **alternateGatekeeper** list in an RCF from the gatekeeper when it registered or in a previous LRJ message.

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

integrityCheckValue – Provides improved message integrity/message authentication of the RAS messages. The cryptographically-based integrity check value is computed by the sender applying a negotiated integrity algorithm and the secret key upon the entire message. Prior to **integrityCheckValue** computation, this field shall be ignored and shall be empty. After computation, the sender puts the computed integrity check value in the **integrityCheckValue** field and transmits the message.

desiredProtocols – Identifies the type of protocols, in order of preference, the originating endpoint desires for its call (e.g., voice or fax). A resolving entity may use this field to locate an endpoint that also supports the protocol, giving consideration to the order of preference.

desiredTunnelledProtocol – This field identifies a protocol that is requested to be tunnelled.

featureSet – This field specifies a set of generic features that relate to this call.

genericData – This field is a list of generic elements related to features that are defined outside of the base H.225.0 specification. These parameters may be used, for example, for tunnelling information transparently through RAS.

hopCount – This field defines the number of gatekeepers through which this message may propagate. When a gatekeeper receives an LRQ and decides that the message should be forwarded on to another gatekeeeper, it first decrements **hopCount**. If **hopCount** is then greater than 0, the gatekeeper inserts the new hop count value into the message to be forwarded. If **hopCount** has reached 0, the gatekeeper shall not forward the message.

circuitInfo – This field provides information about the SCN circuit or circuits used for this call.

callIdentifier – A globally unique call identifier set by the originating endpoint which can be used to associate RAS signalling with the Call Control signalling used in this Recommendation. When sending an LRQ in support of an ARQ or SETUP, the gatekeeper shall copy the call identifier from the ARQ or SETUP into the LRQ. An endpoint that sends an LRQ in preparation for initiating a call shall populate this field with the call identifier for the call. LRQs sent outside the context of a call will not include the call identifier field.

bandWidth – The bidirectional bandwidth requested for the call, in units of 100 bits per second. For example, a 128 kbit/s call would be signalled as a request for 256 kbit/s. The value refers only to the audio and video bit rate excluding headers and overhead.

sourceEndpointInfo – Sequence of alias addresses for the source endpoint, such as **dialledDigits**, **PartyNumber** (**e164Number** or **privateNumber**) or **h323-ID**s. The gatekeeper should copy the information for the endpoint on whose behalf it is sending this LRQ, or if, forwarding a received LRQ, the gatekeeper should copy the **sourceEndpointInfo** from the received LRQ.

canMapSrcAlias – If set to TRUE, indicates that if the resulting LCF contains **modifiedSrcInfo**, the endpoint can copy this information to the **sourceInfo** field of the Setup message. If the LRQ is being sent by a gatekeeper as a result of receiving an ARQ, the gatekeeper shall copy this field from the ARQ. If the gatekeeper would replace addressing information from the LRQ and **canMapSrcAlias** is FALSE, then the gatekeeper should reject the LRQ.

7.13.2 LocationConfirm (LCF)

The LCF message includes the following:

requestSeqNum – This shall be the same value that was passed in the LRQ.

callSignalAddress – The transport address to which to send H.225.0 call signalling; uses the reliable well known or dynamic port, but may be an endpoint or gatekeeper address depending on the call model in use.

rasAddress – Registration, admissions, and status address for the located endpoint.

nonStandardData – Carries information not defined in this Recommendation (for example, proprietary data).

destinationInfo – Sequence of alias addresses for the destination, such as **dialledDigits**, **partyNumber** (**e164Number** or **privateNumber**) or **h323-ID**s.

destExtraCallInfo – Contains external addresses for multiple calls.

destinationType – This specifies the type of the destination endpoint.

remoteExtensionAddress – Contains the alias address of a called endpoint in cases where this information is needed to traverse multiple Gateways.

alternateEndpoints – A sequence of prioritized endpoint alternatives for **callSignalAddress**, **rasAddress**, or **destinationInfo**.

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

integrityCheckValue – Provides improved message integrity/message authentication of the RAS messages. The cryptographically-based integrity check value is computed by the sender applying a negotiated integrity algorithm and the secret key upon the entire message. Prior to **integrityCheckValue** computation, this field shall be ignored and shall be empty. After computation, the sender puts the computed integrity check value in the **integrityCheckValue** field and transmits the message.

alternateTransportAddresses – This field conveys call signalling addresses for transports other than TCP. Inclusion of an address indicates support for the corresponding transport.

supportedProtocols – This field indicates the protocols supported by the endpoint.

multipleCalls – If TRUE, this field indicates that the located endpoint is capable of signalling multiple calls over a single call signalling connection. If FALSE, the endpoint does not have this capability. If this field is not present, the gatekeeper does not know whether the endpoint has this capability.

featureSet – This field specifies a set of generic features that relate to this call.

genericData – This field is a list of generic elements related to features that are defined outside of the base H.225.0 specification. These parameters may be used, for example, for tunnelling information transparently through RAS.

circuitInfo – This field provides information about the SCN circuit or circuits used for this call.

serviceControl – This field contains addressing information that the endpoint may use for call-related service control communication with the network as described, for example, in Annex K/H.323.

modifiedSrcInfo – Alias address that should be used for the source endpoint, such as **dialledDigits**, **PartyNumber** (**e164Number** or **privateNumber**) or **h323-ID**s. This field should be used when the calling endpoint's alias address is translated/modified when attempting to route the call to the primary destination, or to any of the alternate endpoints. If the LCF message results in an ACF reply to the endpoint, this field shall be copied into the ACF message.

bandWidth – The allowed maximum bandwidth for the call; may be less than that requested.

7.13.3 LocationReject (LRJ)

The LRJ message includes the following:

requestSeqNum – This shall be the same value that was passed in the LRQ.

rejectReason – This is the reason the location request was denied. If **rejectReason** is **routeCallToSCN**, the **rejectReason** for this choice also includes a telephone number, or list of telephone numbers, to which the gateway can redirect the call in the SCN, if the gateway supports such a procedure. A reason of **resourceUnavailable** indicates that bandwidth is over utilized or that no entity registered with the gatekeeper has the capacity to handle a call to the requested location at the present time. A reason of **genericDataReason** indicates that the request was rejected as a result of a generic element or feature; in this case, additional information may be specified in the **genericData** field.

nonStandardData – Carries information not defined in this Recommendation (for example, proprietary data).

altGKInfo – Optional information about alternative gatekeepers.

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

integrityCheckValue – Provides improved message integrity/message authentication of the RAS messages. The cryptographically-based integrity check value is computed by the sender applying a negotiated integrity algorithm and the secret key upon the entire message. Prior to **integrityCheckValue** computation, this field shall be ignored and shall be empty. After computation, the sender puts the computed integrity check value in the **integrityCheckValue** field and transmits the message.

featureSet – This field specifies a set of generic features that relate to this call.

genericData – This field is a list of generic elements related to features that are defined outside of the base H.225.0 specification. These parameters may be used, for example, for tunnelling information transparently through RAS.

serviceControl – This field contains addressing information that the endpoint may use for call-related service control communication with the network as described, for example, in Annex K/H.323.

7.14 Disengage messages

7.14.1 DisengageRequest (DRQ)

If sent from an endpoint to a gatekeeper, the DRQ informs the gatekeeper that an endpoint is being dropped. If sent from a gatekeeper to an endpoint, the DRQ forces a call to be dropped; such a request shall not be refused. The DRQ is not sent between endpoints directly.

Note that DRQ is not the same as **ReleaseComplete** since its purpose is to inform the gatekeeper of the termination of a call; the gatekeeper may not receive the release complete if it is not terminating the call signalling channel.

The DRQ message includes the following:

requestSeqNum – This is a monotonically increasing number unique to the sender. It shall be returned by the receiver in any messages associated with this specific message.

endpointIdentifier – This is an endpoint identifier that was assigned to the terminal by RCF.

conference ID – ID of the call that is to have the bandwidth released.

callReferenceValue – The CRV from H.225.0 call signalling messages for this call; only local validity. This is used by a gatekeeper to associate the message with a particular call.

disengageReason – The reason the change was requested by the gatekeeper or the terminal.

nonStandardData – Carries information not defined in this Recommendation (for example, proprietary data).

callIdentifier – A globally unique call identifier set by the originating endpoint which can be used to associate RAS signalling with the modified Q.931 signalling used in this Recommendation.

gatekeeperIdentifier – A **gatekeeperIdentifier** which the endpoint received in the **alternateGatekeeper** list in an RCF from the gatekeeper when it registered or in a previous DRJ message.

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

integrityCheckValue – Provides improved message integrity/message authentication of the RAS messages. The cryptographically-based integrity check value is computed by the sender applying a negotiated integrity algorithm and the secret key upon the entire message. Prior to **integrityCheckValue** computation, this field shall be ignored and shall be empty. After computation, the sender puts the computed integrity check value in the **integrityCheckValue** field and transmits the message.

answeredCall – Set to TRUE to indicate that this party was the original destination (this party answered the call).

callLinkage – The contents of this field are typically controlled by a call linkage service. For the procedures and semantics of this field, refer to clause 10/H.323.

capacity – This field indicates the sending endpoint's available call capacity at this point in time, assuming that the gatekeeper confirms the DRQ by sending a DCF. When sending this field, the endpoint shall include the **currentCallCapacity** element. This field is not included when the DRQ is sent by a gatekeeper.

circuitInfo – This field provides information about the SCN circuit or circuits used for this call.

usageInformation – This field allows an endpoint to report usage information for this call. A gatekeeper shall not include this field when sending a DRQ.

terminationCause – This field describes the reason that the call ended. This information is more specific than the reason provided in the **disengageReason** field. A gatekeeper shall not include this field when sending a DRQ.

serviceControl – This field contains service-specific data, or references to it, that could be used by an endpoint as described, for example, in Annex K/H.323. The gatekeeper could use this field to indicate that the call is ending because some account has expired or the amount paid for the call has been exhausted.

genericData – This field is a list of generic elements related to features that are defined outside of the base H.225.0 specification. These parameters may be used, for example, for tunnelling information transparently through RAS.

7.14.2 DisengageConfirm (DCF)

The DCF message includes the following:

requestSeqNum – This shall be the same value that was passed in the DRQ.

nonStandardData – Carries information not defined in this Recommendation (for example, proprietary data).

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

integrityCheckValue – Provides improved message integrity/message authentication of the RAS messages. The cryptographically-based integrity check value is computed by the sender applying a negotiated integrity algorithm and the secret key upon the entire message. Prior to **integrityCheckValue** computation, this field shall be ignored and shall be empty. After computation, the sender puts the computed integrity check value in the **integrityCheckValue** field and transmits the message.

capacity – This field indicates the sending endpoint's available call capacity after the call indicated in the DCF has been disengaged. When sending this field, the endpoint shall include the **currentCallCapacity** element. This field is not included when the DCF is sent by a gatekeeper.

circuitInfo – This field provides information about the SCN circuit or circuits used for this call.

usageInformation – This field allows an endpoint to report usage information for this call. A gatekeeper shall not include this field when sending a DCF.

genericData – This field is a list of generic elements related to features that are defined outside of the base H.225.0 specification. These parameters may be used, for example, for tunnelling information transparently through RAS.

7.14.3 DisengageReject (DRJ)

DRJ is sent by the gatekeeper if the endpoint is unregistered.

The DRJ message includes the following:

requestSeqNum – This shall be the same value that was passed in the DRQ.

rejectReason – The reason the request was rejected.

nonStandardData – Carries information not defined in this Recommendation (for example, proprietary data).

altGKInfo – Optional information about alternative gatekeepers.

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

integrityCheckValue – Provides improved message integrity/message authentication of the RAS messages. The cryptographically-based integrity check value is computed by the sender applying a negotiated integrity algorithm and the secret key upon the entire message. Prior to **integrityCheckValue** computation, this field shall be ignored and shall be empty. After computation, the sender puts the computed integrity check value in the **integrityCheckValue** field and transmits the message.

genericData – This field is a list of generic elements related to features that are defined outside of the base H.225.0 specification. These parameters may be used, for example, for tunnelling information transparently through RAS.

7.15 Status request messages

The IRQ is sent from a gatekeeper to a terminal requesting status information in the form of an IRR. The IRR may be also be sent by the terminal at an interval specified in the ACF message without the receipt of an IRQ from the gatekeeper. This message should not be confused with the H.225.0 call signalling Status message.

When an unsolicited IRR is sent by an endpoint to a gatekeeper of version 2 or higher, it may indicate in the **needResponse** field that it wishes the gatekeeper to acknowledge receipt of the IRR. In this case it fills in the **requestSeqNum** field with a number other than 1. The gatekeeper returns either an IACK (positive acknowledgement) or an INAK (negative acknowledgement) message, and shall return the same number in the **requestSeqNum** field.

7.15.1 InfoRequest (IRQ)

The IRQ message includes the following:

requestSeqNum – This is a monotonically increasing number unique to the sender. It shall be returned by the receiver in any messages associated with this specific message.

callReferenceValue – CRV of the call that the query is about. If zero, this message is interpreted as a request for an IRR for each call the terminal is active on. If the terminal is not active on any calls, an IRR shall be sent in response to a **callReferenceValue** of 0 with all appropriate fields provided. If **callReferenceValue** is 0, the endpoint shall ignore **callIdentifier** – in this case the gatekeeper shall fill **callIdentifier** with 0.

nonStandardData – Carries information not defined in this Recommendation (for example, proprietary data).

replyAddress – A transport address to send IRR to, perhaps not that of the gatekeeper.

callIdentifier – A globally unique call identifier set by the originating endpoint which can be used to associate RAS signalling with the modified Q.931 signalling used in this Recommendation.

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

integrityCheckValue – Provides improved message integrity/message authentication of the RAS messages. The cryptographically-based integrity check value is computed by the sender applying a negotiated integrity algorithm and the secret key upon the entire message. Prior to **integrityCheckValue** computation, this field shall be ignored and shall be empty. After computation, the sender puts the computed integrity check value in the **integrityCheckValue** field and transmits the message.

uuiesRequested – The gatekeeper may request the endpoint to notify the gatekeeper of H.225.0 call signalling messages that the endpoint sends or receives if the endpoint indicated this capability in the ARQ by setting **willSupplyUUIEs** to TRUE. **uuiesRequested** indicates the set of H.225.0 call signalling messages of which the endpoint shall notify the gatekeeper.

callLinkage – The contents of this field are typically controlled by a call linkage service. For the procedures and semantics of this field, refer to clause 10/H.323.

usageInfoRequested – This field may be included by a gatekeeper to request that the endpoint report the indicated call usage information in the IRR message.

segmentedResponseSupported – This field indicates whether the gatekeeper will allow the endpoint to return call information for all calls in multiple IRR messages, or "segments". If this field is present, segmentation is allowed. Otherwise, segmentation is not allowed. This field is only significant when the gatekeeper sends an IRQ with a **callReferenceValue** of 0 and shall not be present otherwise.

nextSegmentRequested – If the gatekeeper sends an IRQ message with **callReferenceValue** of 0 and includes the **segmentedResponseSupported** field, the endpoint may return an IRR with only part of the call information, indicated by including the segment field in the IRR. The gatekeeper may request the next segment by retransmitting the previous IRQ message with the **nextSegmentRequested** field set to the value of the next segment that the gatekeeper expects to receive.

capacityInfoRequested – If present, this field indicates that the gatekeeper is requesting that the endpoint include call capacity information in the IRR.

genericData – This field is a list of generic elements related to features that are defined outside of the base H.225.0 specification. These parameters may be used, for example, for tunnelling information transparently through RAS.

7.15.2 InfoRequestResponse (IRR)

The IRR message includes the following:

nonStandardData – Carries information not defined in this Recommendation (for example, proprietary data).

requestSeqNum – In the case of a solicited IRR, this field shall contain the sequence number from the IRQ. In the case of an unsolicited report to a version 1 gatekeeper, this field shall contain one (1). In all other unsolicited IRRs, it shall contain a monotonically increasing number (to be returned by the gatekeeper in its response if **needResponse** is TRUE).

endpointType – Provides information about the endpoint.

endpointIdentifier – Value assigned by the gatekeeper in the RCF.

rasAddress – Address for registration, admissions, etc.

callSignalAddress – Address of H.225.0 call signalling.

endpointAlias – Alias(es) for endpoint.

perCallInfo – Information about a particular call:

- **nonStandardData** – Carries information not defined in this Recommendation (for example, proprietary data).
- **callReferenceValue** – H.225.0 call signalling CRV of that call that the response is about.
- **conferenceID** – Unique conference identifier.
- **originator** – If TRUE the endpoint being queried was the call originator, if FALSE the endpoint was the call destination.
- **audio** – Information about the audio channel(s). The **multicast** element shall be included if the session is multicast.
- **video** – Information about the video channel(s). The **multicast** element shall be included if the session is multicast.
- **data** – Information about the data channel(s).
- **h245** – The transport address of the H.245 control channel.
- **callSignalling** – The transport address of the H.225.0 call signalling channel.
- **callType** – Provides information on call topology.

- **bandwidth** – Current usage in increments of 100 bit/s; includes only audio and video excluding headers and overhead.
- **callModel** – Indicates the endpoint's idea of which call model is in use.
- **callIdentifier** – A globally unique call identifier set by the originating endpoint which can be used to associate RAS signalling with the modified Q.931 signalling used in this Recommendation.
- **tokens** – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.
- **cryptoTokens** – Encrypted **tokens**.
- **substituteConfIDs** – A listing of all ConferenceIDs received in H.245 SubstituteCID messages pertaining to the original RAS **perCallInfo conferenceID**.
- **pdu**:
  - **h323pdu** – A copy of an H.225.0 and Q.931 PDU as requested by the gatekeeper in **uuiesRequested** in either ACF or IRQ.
  - **sent** – Set to TRUE to indicate the endpoint sent the **h323pdu**; set to FALSE to indicate the endpoint received the **h323pdu**.
- **callLinkage** – The contents of this field are typically controlled by a call linkage service. For the procedures and semantics of this field, refer to clause 10/H.323.
- **usageInformation** – This field allows the endpoint to report usage information for this call.
- **circuitInfo** – This field provides information about the SCN circuit or circuits used for this call.

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

integrityCheckValue – Provides improved message integrity/message authentication of the RAS messages. The cryptographically-based integrity check value is computed by the sender applying a negotiated integrity algorithm and the secret key upon the entire message. Prior to **integrityCheckValue** computation, this field shall be ignored and shall be empty. After computation, the sender puts the computed integrity check value in the **integrityCheckValue** field and transmits the message.

needResponse – If this is set to TRUE and the gatekeeper indicated in either RCF or ACF that it will respond to unsolicited IRRs (by setting **willRespondToIRR** to TRUE), then the gatekeeper shall reply with an IACK or INAK. If the gatekeeper had not indicated in either RCF or ACF that it will respond to unsolicited IRRs (by setting **willRespondToIRR** to FALSE), then the gatekeeper may ignore the **needResponse** BOOLEAN.

capacity – Indicates the sending endpoint's call capacity at this point in time. When sending this field, the endpoint shall include the **currentCallCapacity** element and should only include the **maximumCallCapacity** when responding to an IRQ that included the **capacityInfoRequested** element.

irrStatus – This element should be returned in IRR messages in response to an IRQ sent by the gatekeeper. Absence of this element indicates that the IRR message contains complete call detail information. The following values are possible:

- **complete** – Indicates that this IRR contains the last segment of call information for an IRQ which requests all call details. When segmentation is not used, this field indicates that the IRR contains all of the call details in a single IRR message.

- **incomplete** – Indicates that the endpoint is not able to fit all of the requested call information in a single IRR message when responding to an IRQ message that contained a **callReferenceValue** of 0.
- **segment** – This field indicates the segment number, which is a monotonically increasing value modulo 65536, of this IRR message when segmented IRRs are sent in response to an IRQ containing a **callReferenceValue** of 0.
- **invalidCall** – This field indicates that the call referenced in the IRQ message does not exist.

unsolicited – H.323 version 4 and later endpoints shall set this field to TRUE in unsolicited IRR messages as described in 8.4.2/H.323 and shall set it to FALSE it in solicited IRRs.

genericData – This field is a list of generic elements related to features that are defined outside of the base H.225.0 specification. These parameters may be used, for example, for tunnelling information transparently through RAS.

7.15.3 InfoRequestAck (IACK)

The IACK message includes the following:

requestSeqNum – This field shall contain the **requestSeqNum** that was in the IRR.

nonStandardData – Carries information not defined in this Recommendation (for example, proprietary data).

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

integrityCheckValue – Provides improved message integrity/message authentication of the RAS messages. The cryptographically-based integrity check value is computed by the sender applying a negotiated integrity algorithm and the secret key upon the entire message. Prior to **integrityCheckValue** computation, this field shall be ignored and shall be empty. After computation, the sender puts the computed integrity check value in the **integrityCheckValue** field and transmits the message.

7.15.4 InfoRequestNak (INAK)

The INAK message includes the following:

requestSeqNum – This field shall contain the **requestSeqNum** that was in the IRR.

nonStandardData – Carries information not defined in this Recommendation (for example, proprietary data).

nakReason – Reason the IRR was negatively acknowledged.

altGKInfo – Optional information about alternative gatekeepers.

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

integrityCheckValue – Provides improved message integrity/message authentication of the RAS messages. The cryptographically-based integrity check value is computed by the sender applying a negotiated integrity algorithm and the secret key upon the entire message. Prior to **integrityCheckValue** computation, this field shall be ignored and shall be empty. After computation, the sender puts the computed integrity check value in the **integrityCheckValue** field and transmits the message.

7.16 Non-standard message

The **NonStandardMessage** structure is as follows:

requestSeqNum – This is a monotonically increasing number unique to the sender.

nonStandardData – Carries information not defined in this Recommendation (for example, proprietary data).

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

integrityCheckValue – Provides improved message integrity/message authentication of the RAS messages. The cryptographically-based integrity check value is computed by the sender applying a negotiated integrity algorithm and the secret key upon the entire message. Prior to **integrityCheckValue** computation, this field shall be ignored and shall be empty. After computation, the sender puts the computed integrity check value in the **integrityCheckValue** field and transmits the message.

featureSet – This field specifies a set of generic features.

genericData – This field is a list of generic elements related to features that are defined outside of the base H.225.0 specification. These parameters may be used, for example, for tunnelling information transparently through RAS.

7.17 Message not understood

This message is sent whenever an H.323 endpoint receives a RAS message it does not understand or it cannot decode. In cases where the destination transport address for the XRS message is not available (i.e., the received RAS message could not be decoded), the XRS may be sent to the transport address from which the not understood RAS message was received. This transport address may be obtained from the underlying transport layer. An XRS message shall not be sent in response to an incoming XRS message. H.323 endpoints should transmit no more than one XRS message per second to the same transport address to avoid network congestion in situations where corrupted messages are received.

RequestSeqNum – Shall be the **requestSeqNum** of the unknown message, if it can be decoded. If the unknown message cannot be decoded, this field is a monotonically increasing number unique to the sender. The **RequestSeqNum** should be used for backward compatibility with H.323 version 3 and lower endpoints. H.323 version 4 and higher endpoints should look at the **messageNotUnderstood** parameter to associate the XRS with a previously transmitted message.

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

integrityCheckValue – Provides improved message integrity/message authentication of the RAS messages. The cryptographically-based integrity check value is computed by the sender applying a negotiated integrity algorithm and the secret key upon the entire message. Prior to **integrityCheckValue** computation, this field shall be ignored and shall be empty. After computation, the sender puts the computed integrity check value in the **integrityCheckValue** field and transmits the message.

messageNotUnderstood – Copy of the message that was received and was not understood.

7.18 Gateway resource availability messages

The Resource Availability Indication (RAI) is a notification from a gateway to a gatekeeper of its current call capacity for each H-series protocol and data rate for that protocol. The gatekeeper responds with a Resource Availability Confirmation (RAC) upon receiving a RAI to acknowledge its reception.

7.18.1 ResourcesAvailableIndicate (RAI)

The RAI message includes the following:

requestSeqNum – This is a monotonically increasing number unique to the sender. It shall be returned by the receiver in any response associated with this specific message.

protocolIdentifier – Identifies the vintage of the sending endpoint.

nonStandardData – Carries information not defined in this Recommendation (for example, proprietary data).

endpointIdentifier – A gatekeeper-assigned endpoint identity string.

protocols – Indicates the current data rates for each protocol which can be supported given the current state of the device.

almostOutOfResources – When set to TRUE, the device is nearing or at capacity. Any action based on this field is at the manufacturer's discretion. If the device is not near or at capacity this field should be set to FALSE.

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

integrityCheckValue – Provides improved message integrity/message authentication of the RAS messages. The cryptographically-based integrity check value is computed by the sender applying a negotiated integrity algorithm and the secret key upon the entire message. Prior to **integrityCheckValue** computation, this field shall be ignored and shall be empty. After computation, the sender puts the computed integrity check value in the **integrityCheckValue** field and transmits the message.

capacity – Indicates the sending endpoint's call capacity at this point in time. Note that if **capacity** is provided, the **almostOutOfResources** BOOLEAN should be ignored by the recipient, since the **capacity** field provides more detailed information; however, the **almostOutOfResources** BOOLEAN shall be properly set in order to maintain backward compatibility. When sending the **capacity** field, the endpoint shall include the **currentCallCapacity** elements.

genericData – This field is a list of generic elements related to features that are defined outside of the base H.225.0 specification. These parameters may be used, for example, for tunnelling information transparently through RAS.

7.18.2 ResourcesAvailableConfirm (RAC)

The RAC message includes the following:

requestSeqNum – This shall be the same value that was passed in the RAI.

protocolIdentifier – Identifies the vintage of the accepting gatekeeper.

nonStandardData – Carries information not defined in this Recommendation (for example, proprietary data).

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

integrityCheckValue – Provides improved message integrity/message authentication of the RAS messages. The cryptographically-based integrity check value is computed by the sender applying a negotiated integrity algorithm and the secret key upon the entire message. Prior to **integrityCheckValue** computation, this field shall be ignored and shall be empty. After computation, the sender puts the computed integrity check value in the **integrityCheckValue** field and transmits the message.

genericData – This field is a list of generic elements related to features that are defined outside of the base H.225.0 specification. These parameters may be used, for example, for tunnelling information transparently through RAS.

7.19 RAS timers and Request in Progress (RIP)

Table 23 shows recommended default time-out values for the response to RAS messages and subsequent retry counts if a response is not received. (These values are subject to change with further implementation experience and input.)

Table 23/H.225.0 – Recommended default time-out values

RAS message	Time-out value (s)	Retry count
GRQ	5	2
RRQ (Note 1)	3	2
URQ	3	1
ARQ	5	2
BRQ	3	2
IRQ	3	1
IRR (Note 2)	5	2
DRQ	3	2
LRQ	5	2
RAI	3	2
SCI	3	2
NOTE 1 – The time-out value should be recalculated based upon both the time-to-live (which may be indicated by the gatekeeper in the RCF message) and the desired number of retries. NOTE 2 – In cases where the gatekeeper is expected to reply to an unsolicited IRR with IACK or INAK, the time-out may occur if no reply to the IRR is received.		

If an entity receives a request from a version 2 (or later) entity to which a response cannot be generated within a typical retry time-out period, it can send a RIP message specifying the period (in the **delay** field) after which a response should have been generated. As soon as a response is available, the responding entity should send the response and not wait for the RIP delay to expire. If a requesting entity has not received a response by the time the RIP delay expires, it shall resend the request. The responding entity can then either send a duplicate response or another RIP message. Figure 2 gives an example message exchange which demonstrates a number of aspects of the retry strategy.

Vendors should be aware that any retries will have an impact on the call setup time, which should be minimized. Therefore, short retry times are desirable. So that remote entities can anticipate typical retry times for the purpose of deciding when to send a RIP message, entities should avoid retry periods less than 100 ms. Exponential backoff and adapting to measured round-trip times is encouraged. Entities can use the measured round trip time of the RRQ/RCF registration process to modify an initially conservative estimate (of a few seconds) for this purpose. Entities may also use the registration process to exchange version numbers to ensure that the RIP-based retry mechanism is not used when version 1 entities are involved in the signalling.

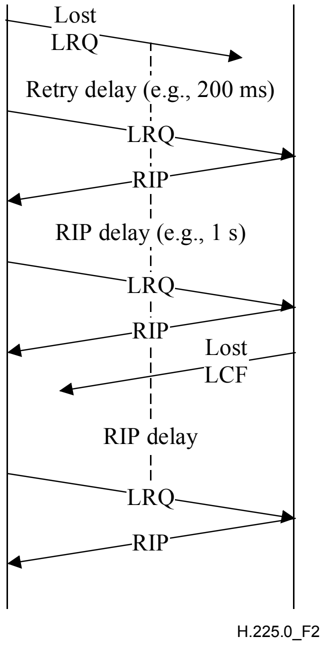


Figure 2/H.225.0 – Example use of RIP message

The RIP message includes the following:

requestSeqNum – This is the **requestSeqNum** of the request which is currently being processed.

nonStandardData – Carries information not defined in this Recommendation (for example, proprietary data).

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

integrityCheckValue – Provides improved message integrity/message authentication of the RAS messages. The cryptographically-based integrity check value is computed by the sender applying a negotiated integrity algorithm and the secret key upon the entire message. Prior to **integrityCheckValue** computation, this field shall be ignored and shall be empty. After computation, the sender puts the computed integrity check value in the **integrityCheckValue** field and transmits the message.

delay – Specifies the amount of time in milliseconds that an endpoint shall wait before attempting a retry. The responding endpoint may respond before this period expires.

7.20 Service control messages

7.20.1 ServiceControlIndication (SCI)

The SCI message is sent from a service provider to indicate to the service client that a separate service control session may be initiated towards the given address. It may be sent from a gatekeeper to an endpoint (e.g., for user presentation of service features) or from an endpoint to a gatekeeper (e.g., to upload a call processing script). Note that H.323 entities of version 3 or earlier are not able to decode this message and thus will not answer.

The SCI message contains the following:

requestSeqNum – This is a monotonically increasing number unique to the sender. It shall be returned by the receiver in any response associated with this specific message.

nonStandardData – Carries information not defined in this Recommendation (for example, proprietary data).

serviceControl – Carries a set of service control session information.

endpointIdentifier – Set to the value received from the gatekeeper in the RCF message if the message is sent from an endpoint to its gatekeeper.

callSpecific – Provided if the sessions given are relating to one specific call. The **callIdentifier**, **conferenceID** and **answeredCall** shall be set to the same value as in the ARQ message the service session is relating to.

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

integrityCheckValue – Provides improved message integrity/message authentication of the RAS messages. The cryptographically-based integrity check value is computed by the sender applying a negotiated integrity algorithm and the secret key upon the entire message. Prior to **integrityCheckValue** computation, this field shall be ignored and shall be empty. After computation, the sender puts the computed integrity check value in the **integrityCheckValue** field and transmits the message.

featureSet – This field specifies a set of generic features.

genericData – This field is a list of generic elements related to features that are defined outside of the base H.225.0 specification. These parameters may be used, for example, for tunnelling information transparently through RAS.

7.20.2 ServiceControlResponse (SCR)

The SCR message is sent to acknowledge the receipt of an SCI message, but does not necessarily mean that the service client will initiate the session as given in SCI.

The SCR message contains the following:

requestSeqNum – This shall be the same value that was passed in the SCI.

result – This field indicates the result of processing the information contained in the SCI message. The following values are defined:

- **started** – The requested service control was started.
- **failed** – There was some error with the request, so the request failed.
- **stopped** – The service control was stopped.
- **notAvailable** – The requested service control was not available at the time of the request.

nonStandardData – Carries information not defined in this Recommendation (for example, proprietary data).

tokens – This is some data which may be required to allow the operation. The data shall be inserted into the message if available.

cryptoTokens – Encrypted **tokens**.

integrityCheckValue – Provides improved message integrity/message authentication of the RAS messages. The cryptographically-based integrity check value is computed by the sender applying a negotiated integrity algorithm and the secret key upon the entire message. Prior to **integrityCheckValue** computation, this field shall be ignored and shall be empty. After computation, the sender puts the computed integrity check value in the **integrityCheckValue** field and transmits the message.

featureSet – This field specifies a set of generic features.

genericData – This field is a list of generic elements related to features that are defined outside of the base H.225.0 specification. These parameters may be used, for example, for tunnelling information transparently through RAS.

7.21 AdmissionConfirmSequence

AdmissionConfirmSequence is a sequence of one or more RAS ACF messages. It may used by the gatekeeper to respond to a single ARQ message instead of the single ACF message when it has different security tokens, different translated source information, etc., that may not easily be expressed in a single ACF message. Endpoints indicate support for receiving AdmissionConfirmSequence by setting **supportsACFSequences** flag in RRQ.

8 Mechanisms for maintaining QoS

8.1 General approach and assumptions

Transport QoS (Quality of Service) on a packet-based network includes such characteristics as:

- bit error rate;
- packet loss rate;
- delay.

Any transport QoS-related signalling (e.g., a reservation request to a router) is done by the terminal as soon as possible, or by the gatekeeper on its behalf. The terminal may wish to make any reservations since the gatekeeper may not be logically near the terminal, or be able to make QoS-related requests on behalf of the terminal. The means by which either the terminal or the gatekeeper make QoS or bandwidth reservations are beyond the scope of this Recommendation.

The Sender and Receiver Reports of RTCP shall be the means by which QoS will be assessed.

There are two types of congestion-related delay that might be measured:

- short-term increases in delay that will result in a perceptible but not annoying slowing of the frame rate;
- a general rise in delay due to packet-based network congestion over time such that a feedback-based mechanism is useful.

Essentially, short-term bursts are approached by error concealment, and a longer term congestion is approached by reducing the multimedia load. The assumption is made that all packet-based network multimedia terminals are H.323 terminals, and all will attempt to reduce packet-based network usage as congestion rises rather than "steal" bandwidth from each other.

Bit errors on a packet-based network generally are either corrected at a lower layer, or result in packet loss, so they are not considered further in this clause.

Packet loss requires the receiver to be able to compensate for lost packets in a fashion that conceals errors to the maximum possible extent. For data and control, retransmission at the transport layer is used. For audio and video, retransmission is for further study.

A given level of transport QoS results in a level of user-perceived audio/video QoS that is a function in part of the effectiveness of the methods used to overcome transport QoS problems.

8.2 Use of RTCP in measuring QoS

8.2.1 Sender reports

The sender report serves three main purposes:

1) allows synchronization of multiple RTP streams, such as audio and video;

2) allows the receiver to know the expected data rate and packet rate;

3) allows the receiver to measure the distance in time to the sender.

Of these three purposes, 1 is the most relevant to this Recommendation. Manufacturers may make use of the sender reports in other ways at their discretion.

The relevant field for stream synchronization is the RTP timestamp and the NTP timestamp in the sender report of RTCP. The NTP timestamp (if available) gives "wall clock" time and corresponds to the RTP timestamp which has the same units and random offset as the RTP capture timestamp in the media packets.

8.2.2 Receiver reports

Four parts of the Receiver Reports are used in this Recommendation to measure QoS:

1) fraction lost;

2) the cumulative packets lost;

3) the extended highest sequence number received;

4) interarrival jitter.

Items 2 and 3 are used to compute the number of packets lost since the previous receiver report. This can be taken as a long-term measure of packet-based network congestion. See RFC 3550 [37] section 6.4.4 for a sample computation. If this loss rate exceeds a value set by the manufacturer, the H.225.0 terminal should reduce the media rates on the packet-based network side according to the procedures in 8.4. If item 1 exceeds a value set by the manufacturer, it may also be desirable to take corrective action.

If the interval between receiver reports exceeds a value set by the manufacturer, H.323 terminals should use item 1 as an indicator of serious congestion requiring media rate reduction on the packet-based network side.

Item 4 should be used as an indication of impending congestion. If interarrival jitter increases for three consecutive receiver reports, the H.323 sending terminal should take corrective action.

8.3 Audio/Video jitter procedures

ITU-T Rec. H.245 provides commands and procedures for round-trip indications using **RoundTripDelayRequest** and **RoundTripDelayResponse**. On a multipoint call the MC responds to a request from the endpoint. RTCP contains a method of calculating round-trip delays based on the Sender Report and the Receiver Report messages. Note that the quantity being measured in each case is not the same, so there is no conflict in using both methods to measure jitter.

See 6.2.5/H.323 for a discussion of how H.245 level signalling can be used to optionally reduce jitter related delays.

8.4 Audio/Video skew procedures

See 6.2.6/H.323 for a discussion of how H.245 level signalling is used to limit the skew between different logical channels.

8.5 Procedures for maintaining QoS

A number of methods exist for the H.323 gateway/terminal to respond to an increase in packet loss or interarrival jitter in the far-end receiver. These methods can be grouped into those that are appropriate for a rapid response to a short-term problem, such as a lost or delayed packet, and those that are appropriate for a response to a longer-term problem such as growing congestion on the packet-based network. Note that these methods do not seek to maintain the current quality of service, but instead, to provide for an orderly degradation of service. The following priorities shall be observed such that, if present, media shall be degraded in the following order: Video, Data, Audio, Control.

Short-term responses

- reducing the frame rate for a short period of time: This may result in the H.323 gateway sending additional H.261 fill frames in the packet-based network to SCN direction to compensate for the packet under flow;
- reduce packet rate by switching to the optional mode where audio/video are mixed in one packet (for further study);
- packet rate can also be reduced via the use of MB fragmentation of the video stream.

Longer-term responses

- reducing media bit rate (e.g., switching from 384 kbit/s to 256 kbit/s): This may involve a simple instruction to the encoder in a terminal, or it may involve the use of a rate reducer function in the H.323 gateway. These changes are signalled via H.245 **FlowControl** commands, or by logical channel signalling as appropriate;
- turning off media of lesser importance (e.g., turning off video to allow a large amount of T.120 traffic);
- returning a busy signal (adaptive busy) to the receiver as an indication of packet-based network congestion. This may be combined with turning off a media, or even all media other than the control Transport Port. Adaptive busy is signalled via a Q.931 cause value in Release Complete.

It should be noted that responding to interarrival jitter in a multi-router path where a large percentage of packets arrive out of order is difficult. It may be impossible to distinguish this source of jitter from other sources, or to base error recovery strategy on measured jitter. However, packet loss is quantifiable and unambiguous.

8.6 Echo control

Control of acoustic echo is the responsibility of the H-series terminal. In general, given the delay involved in video/audio compression, it is assumed that all H.320, H.323 and H.324 terminals have some form of echo control (cancellation or switching).

However, when the H.323 terminal is on a call with a GSTN telephone, it is typically the case that the GSTN phone does not support echo control. Thus, the user of the H.323 terminal may hear acoustic echo return from the GSTN side. This acoustic echo return can be minimized by the use of a speakerphone with echo control, or the use of a handset or ear phones. Manufacturers may add loss to the audio path when an H.323 terminal is connected to a GSTN POTS phone.

Control of hybrid (2- to 4-wire) echo. The hybrid circuit provides an interface between 4-wire transmission systems and 2-wire terminals. For ISDN speech calls that are carried through the GSTN at 64 kbit/s, echo cancellation is not required. For 64 kbit/s data calls, echo cancellation is not permitted.

In the case of a decomposed gateway interfacing to an SS7 network, indications of the provision of echo cancellation are carried in the ISUP signalling message, as specified in ITU-T Rec. Q.115. The H.323 media gateway controller (MGC) can interpret the signalling information and either enable or disable echo cancellation at the media gateway (MG). For speech calls the MGC can enable echo cancellation without deleterious effects on speech quality even if the GSTN has provided echo cancellation in the GSTN.

For voiceband data calls (modem calls) that transit or terminate on an H.323 network, control of echo cancellation is provided by the modems by in-band tones. No out-of-band signalling is required by the GSTN network elements or by the MGCs.

Annex A

RTP/RTCP

RTP and RTCP are defined in reference [37]. This reference is also referred to in Appendix I. Both this annex and Appendix I are being retained in this Recommendation for maintaining equivalence with earlier versions of this Recommendation.

The reader should note that all references in [37] are to a bibliography, and are non-normative, with the exception of the reference to ISO/IEC 10646-1, which also appears in the references clause of this Recommendation.

Readers should also note that the terminology used in [37] differs somewhat from that used in ITU-T Rec. H.323 and this Recommendation according to Table A.1.

Table A.1/H.225.0 – Terminology correspondence

H.323 and H.225.0 term	[37] (RTP/RTCP) term
media stream	data
transport address	transport address
packet-based network address	network address
TSAP identifier	port
Annex A	specification or document
shall	must
should	should

It should be further noted that "translators" and "mixers" are not part of the H.323 system. H.323 endpoints such as gateways and MCUs have some of the characteristics of translators and mixers, so this text has been retained as a guide to the implementor. However, support for translators and mixers is not part of H.323, and these clauses shall be considered informative.

Annex B

RTP profile

RTP Profile is defined in reference [38]. This reference ([38]) is also referred to in Appendix II. Both this annex and Appendix II are being retained in this Recommendation for maintaining equivalence with earlier versions of this Recommendation.

See the introduction to Annex A; all the warnings mentioned there apply to this annex as well.

Annex C

RTP payload format for H.261 video streams

RTP Payload format for H.261 video streams is defined in reference [39]. This reference ([39]) is also referred to in Appendix III. Both this annex and Appendix III are being retained in this Recommendation for maintaining equivalence with earlier versions of this Recommendation.

See the introduction to Annex A; all the warnings mentioned there apply to this annex as well.

Annex D

RTP payload format for H.261A video streams

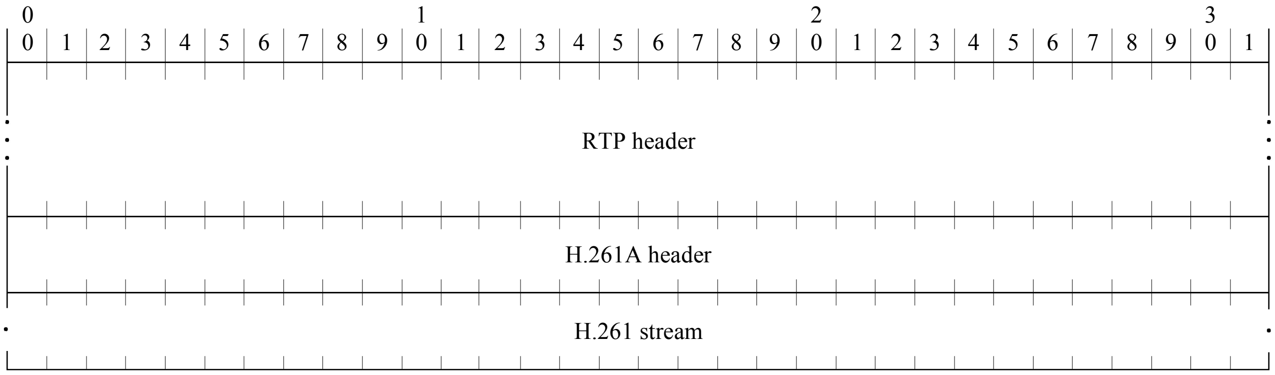
D.1 Introduction

To facilitate interfacing H.323 video streams to the SCN via gateways, ITU-T Rec. H.323 defines a modified form of the RTP H.261 video payload. This eases buffer management and interoperability with remote SCN codecs. Support of the H.261A payload type is signalled using H.245 capability sets and in the **openLogicalChannel** message using RTP dynamic payload types.

D.2 H.261A RTP packetization

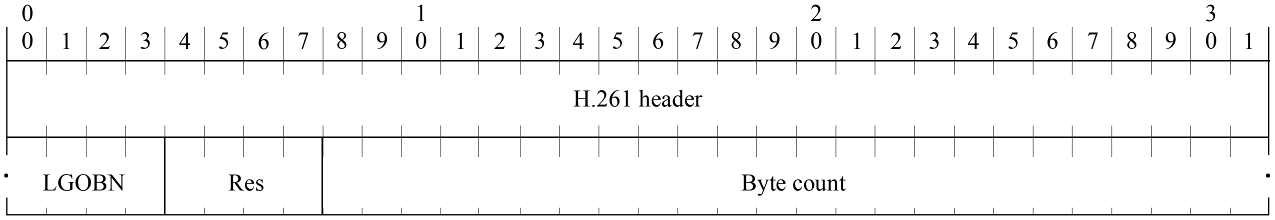
This version is an extension of the version described in Annex C except that an additional 32-bit word is appended to the H.261 header. The procedures that are described in Annex C also apply to this annex.

The H.261A data will follow the RTP header, as in:



H.225.0_FD.2(A)

The H.261A header is defined as:



H.225.0_FD.2(B)

The fields in the H.261A header have the following meanings:

H.261 header: 32 bits – As described in Annex C.

Last GOB Number (LGOBN): 4 bits – The GOB number of the last GOB in the RTP packet (max GOB number is 12 for ITU-T Rec. H.261).

Reserved (RES): Reserved.

Byte Count: 24 bits – Indicates the cumulative number of octets that have been sent in the H.261 stream part of the RTP packets. If the last byte of a packet is only partially filled (as indicated by EBIT), then it is not counted in the cumulative byte count. This modulo 2^{24} byte count starts at a random value and is never reset.

Both of the additional fields may be used when packets are lost or delivered out of order. The Byte Count can be used to determine how much stuffing will be needed in the SCN stream and facilitates buffer management. The last GOB number simplifies determining which GOBs have been lost due to packet loss.

Annex E

Video packetization

This annex describes RTP packetization details for video codecs. Table E.1 provides references to the definitions of video packetization formats that are not defined in this Recommendation. The remaining clauses of this annex define additional video packetization formats.

Table E.1/H.225.0 – Externally defined video packetization formats

Encoding name	Packetization definition
ISO/IEC 14496-2 (MPEG-4 Video)	IETF RFC 3016, *RTP Payload Format for MPEG-4 Audio/Visual Streams*

E.1 H.263

An RTP payload format for H.263 video is specified in IETF RFC 2190 for H.263 video bitstreams that do not contain the new features adopted in version 2 (the 1998 version) of ITU-T Rec. H.263 (the features using PLUSPTYPE or annexes subsequent to Annex H/H.263). An additional payload format which supports the enhanced features of H.263 version 2 bitstreams will be specified at a later date. A legacy packetization format widely used in industry (not as specified in IETF RFC 2190) may only be used if the peer indicated support for this format in the capability exchange.

RFC 3551 [38] section 5 describes the procedure to use to signal H.263 video streams.

Annex F

Audio and multiplexed packetization

This annex describes RTP packetization details for audio codecs. Table F.1 provides references to the definitions of audio packetization formats that are not defined by this Recommendation. Table F.2 provides references to the definitions of multiplexed packetization formats. The remaining clauses of this annex define additional audio packetization formats.

Table F.1/H.225.0 – Externally defined audio packetization formats

Encoding name	Packetization definition
ISO/IEC 14496-3 (MPEG-4 Audio)	IETF RFC 3016, *RTP Payload Format for MPEG-4 Audio/Visual Streams*

Table F.2/H.225.0 – Externally defined multiplexed stream packetization formats

Encoding name	Packetization definition
H.222 multiplexed streams (MPEG-2 transport streams)	IETF RFC 2250, *RTP Payload Format for MPEG1/MPEG2 Video*

F.1 G.723.1

This Recommendation specifies a coded representation that can be used for compressing the speech signal component of multimedia services at a very low bit rate. A G.723.1 frame can be one of three sizes: 24 bytes (6.3 kbit/s frame), 20 bytes (5.3 kbit/s frame), or 4 bytes. These 4-byte frames are called Silence Insertion Descriptor frames (SID) and are used to specify comfort noise parameters. There is no restriction on how 4-, 20-, and 24-byte frames are intermixed. The least significant two bits of the first octet in the frame determine the frame size and codec type (refer to Tables 5 and 6/G.723.1 for more information on bit order). It is possible to switch between the two rates at any 30 ms frame boundary. Both (5.3 kbit/s and 6.4 kbit/s) rates are a mandatory part of the encoder and decoder. This coder was optimized to represent speech with near-toll quality at the above rates using a limited amount of complexity.

All the bits of the encoded bit stream are transmitted always from the least significant bit towards the most significant bit. Note that this refers to the order of bits presented to the transport layer and not the order of bits on the wire.

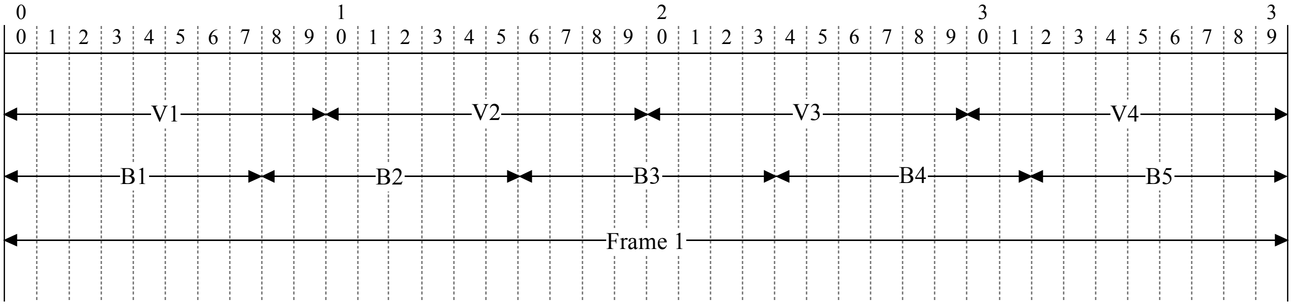
G.723.1 packetization conforms to Annex B except for the packetization interval (30 ms vs. 20 ms default):

1) The first packet of a talkspurt (first packet after a silence period) is distinguished by setting the marker bit in the RTP data header.
2) The sampling frequency (RTP clock frequency) is 8000 Hz.
3) The packetization interval shall have a duration of 30 ms (one frame) as opposed to the default packetization of 20 ms.
4) Codecs should be able to encode and decode several consecutive frames within a single packet.
5) A receiver should accept packets representing between 0 and 180 ms of audio data as opposed to the default of 0 and 200 ms.

F.2 G.728

1) *Frame packetization*

A G.728 frame (4 vectors: V1-V4, 10 bits each, V1 is the older – first to be played) is organized into 5 bytes (B1-B5). Referring to the figure below, the principle for bit order is "maintenance of bit significance". Bits from older vectors are more significant than bits from newer vectors. The Most Significant Bit (MSB) of the frame goes to MSB of B1 and the Least Significant Bit (LSB) of the frame goes to LSB of B5. For clarification: more significant bits from each vector are put in more significant bits of B1-B5 (the more significant bits of lower number B).



H.225.0_FF.0

For example:

B1 contains 8 most significant bits of V1, MSB of V1 is MSB of B1.

B2 contains 2 least significant bits of V2, the more significant of the two in its MSB, and 6 most significant bits of V2, the most significant of them is more significant at B2 also.

B1 shall be put first to the packet (most significant byte in RTP) and B5 last.

2) *Multi-frame packetization*

Ending a single frame in an RTP packet might cause considerable network overhead. Therefore, sending a multi-frame packet is allowed in the following manner:

An RTP G.728 packet shall contain a whole number of frames.

Older frames (to be played first) shall be put first into the RTP packet.

The timestamp would reflect the capturing time of the first sample, in the first vector (V1) of the first frame (the oldest information in the packet).

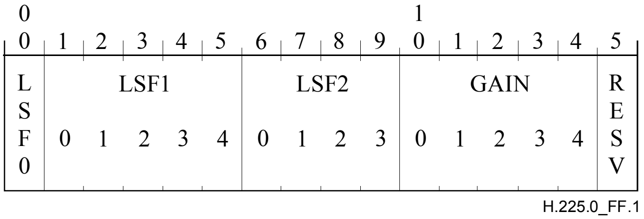
3) The marker bit shall retain the same meaning assigned to it in this Recommendation.

F.3 G.729

This Recommendation specifies a coded representation that can be used for compressing the speech signal component of multi-media services at a bit rate of 8 kbit/s. This coder was optimized to represent speech with toll or wireline quality at 8 kbit/s. This coder has an inherent robustness against random bit errors as well as against randomly and bursty erased frames. It represents speech with a high quality when operating in a noisy environment. A complexity-reduced version of the G.729 algorithm is specified in Annex A/G.729. A floating point version of these two algorithms is specified in Annex C/G.729. The speech coding algorithms in the main body of ITU-T Rec. G.729, and in Annexes A and C/G.729 are fully interoperable with each other, so there is no need to further distinguish between them.

A Voice Activity Detector (VAD) and Comfort Noise Generator (CNG) algorithm in Annex B/G.729 is recommended. This algorithm is applied to Annex F/G.729 (6.4 kbit/s with VAD/CNG), Annex G/G.729 (11.8 kbit/s with VAD/CNG), Annex B/G.729 (G.729 and

Annex A/G.729 with VAD/CNG) and Annex I/G.729. A G.729 or Annex A/G.729 frame contains 10 octets; an Annex D/G.729 frame contains 8 octets; an Annex E/G.729 frame contains 15 octets; and the Annexes B/G.729, F/G.729 and G/G.729 comfort noise frame occupies 2 octets, as shown in Figure F.1.



RESV = Reserved (zero)

Figure F.1/H.225.0 – Annexes B/G.729, F/G.729 and G/G.729 CNG packetization format

The transmitted parameters of a G.729, Annex A/G.729 or Annex C/G.729 10-ms frame, consisting of 80 bits, are defined in Table 8/G.729. The mapping of these parameters is given in Figure F.2. Bits are numbered as Internet order, that is, the most significant bit is bit 0.

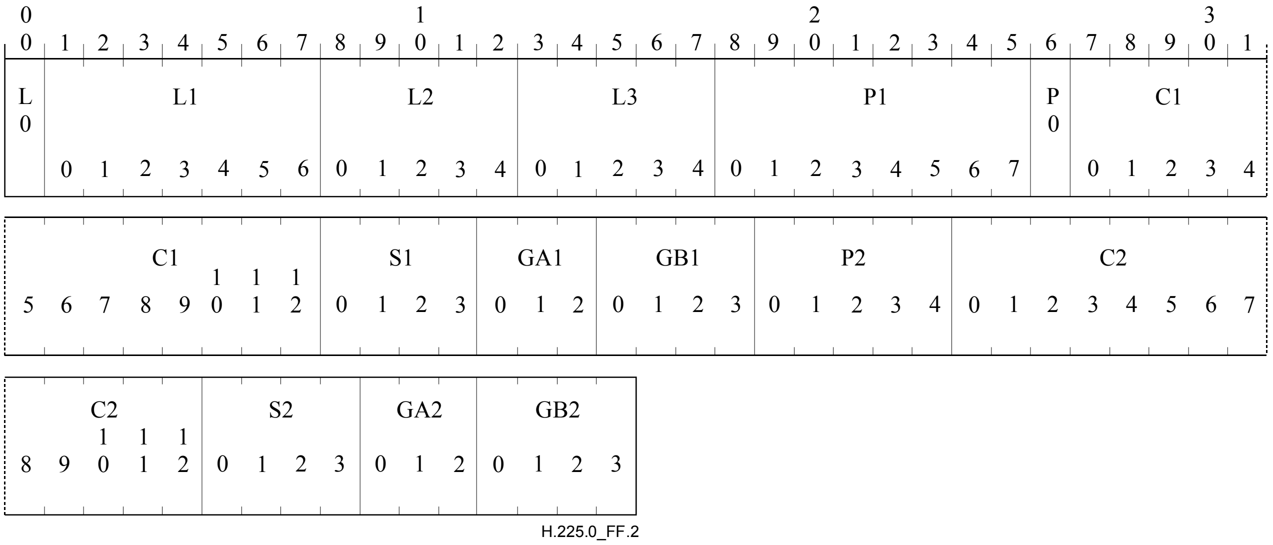


Figure F.2/H.225.0 – G.729, Annex A/G.729 and Annex C/G.729 packetization format

Annex D/G.729 defines a 6.4 kbit/s rate extension of G.729 for momentary reduction in channel capacity, e.g., to handle overload conditions. Annex E/G.729 provides an 11.8 kbit/s extension of G.729 for better performance with a wide range of input signals, such as speech with background noise and music. Additionally, Annex E/G.729 has two operating modes, backward and forward adaptive, which are signalled by the first two bits in the packet header.

The bits of a G.729-6.4 frame are formatted as shown in Figure F.3 (see Table D.1/G.729). Bits are numbered in Internet order; that is, the most significant bit is bit 0. A total of 64 bits are used.

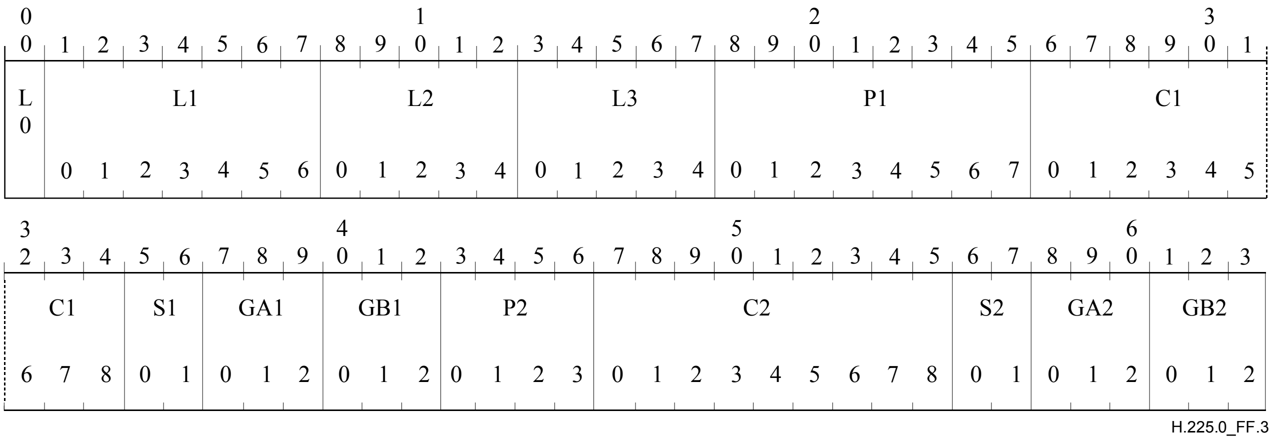


Figure F.3/H.225.0 – G.729-6.4 packetization format

The net bit rate for the Annex E/G.729 algorithm is 11.8 kbit/s and a total of 118 bits are used. The bits of a G.729-12 frame are formatted as shown in Figures F.4 and F.5 (see Table E.1/G.729). Figures F.4 and F.5 describe the fields for the forward adaptive mode and the backward adaptive mode respectively for the Annex E/G.729 algorithm. The two least significant bits are included as "don't care" bits and are used to complete an integer number of octets for the frame.

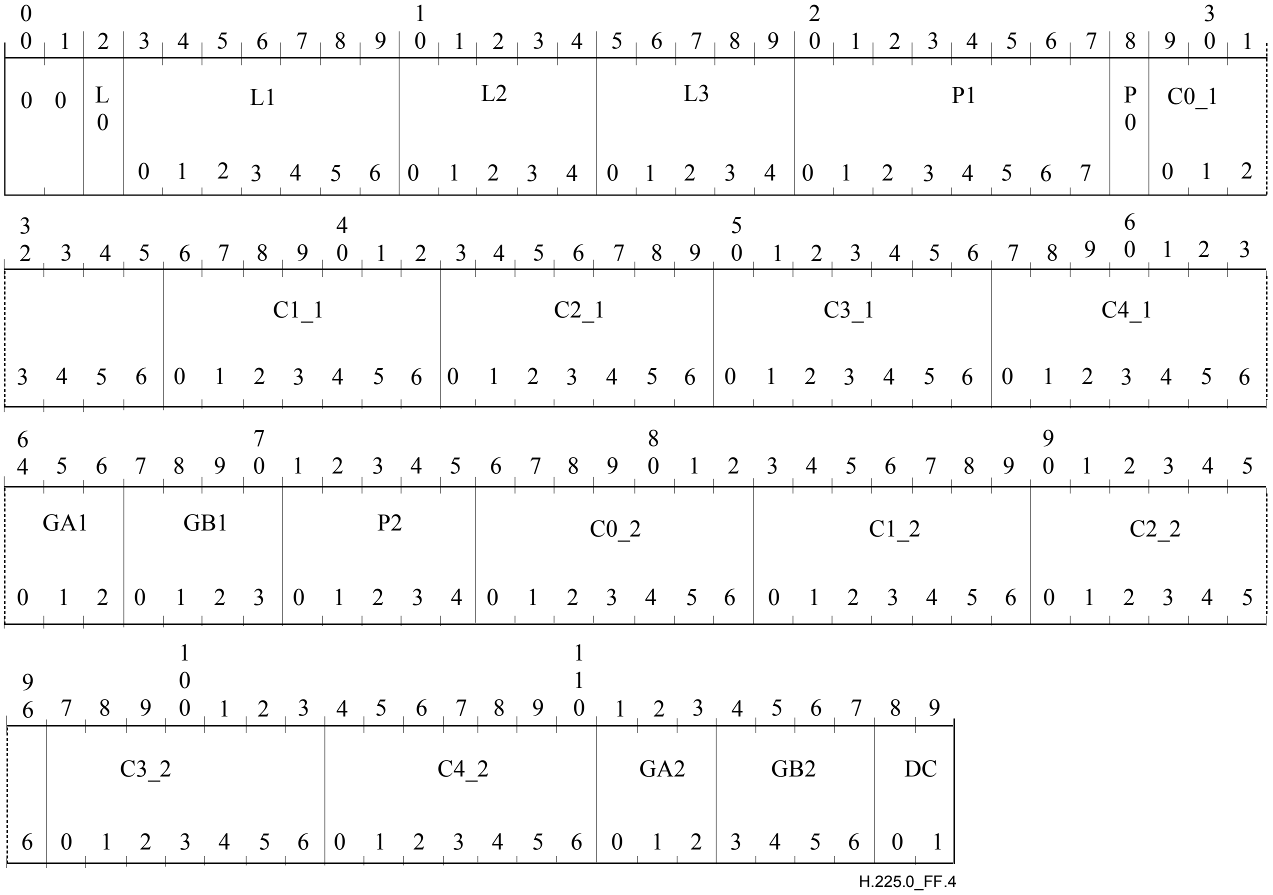


Figure F.4/H.225.0 – G.729-12 packetization format for the forward adaptive mode

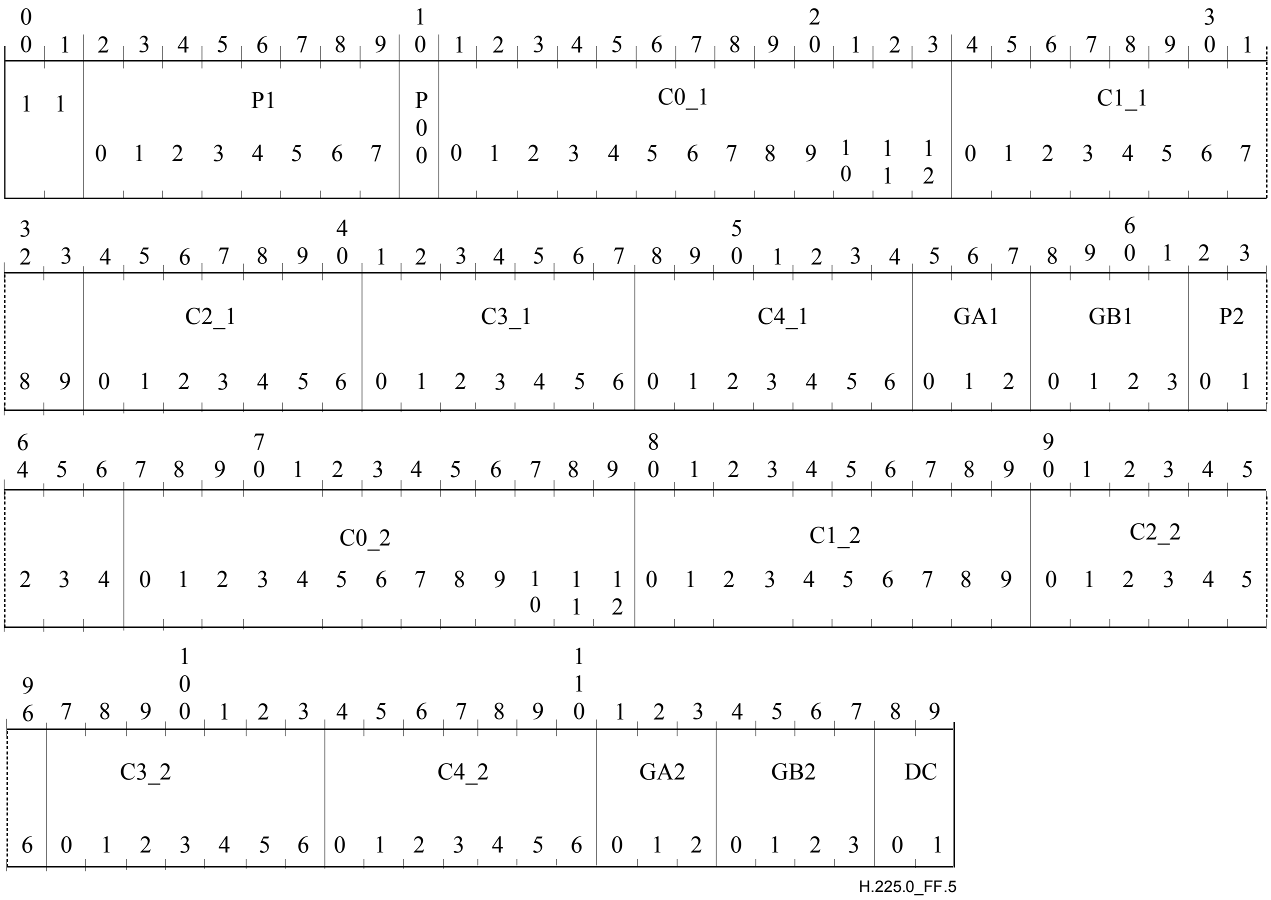


Figure F.5/H.225.0 – G.729-12 packetization format for the backward adaptive mode

An RTP packet may consist of zero or more G.729 or Annex A, C, D or E/G.729 frames, followed by zero or one Annex B/G.729 payloads. The presence of a comfort noise frame can be deduced from the length of the RTP payload.

1) The first packet of a talkspurt (first packet after a silence period) is distinguished by setting the marker bit in the RTP header.

2) The sampling frequency (RTP clock frequency) is 8000 Hz.

3) The default packetization interval should have a duration of 20 ms. While 20 ms is the strongly recommended value, in some situations it may be desirable to send 10-ms packets. For example, consider a transition from voiced to unvoiced in the first 10 ms of the packet. If a 20-ms packetization interval were mandatory, then the transmitter would need to wait until speech is active again.

4) Codecs should be able to encode and decode several consecutive frames within a single packet.

5) A receiver should accept packets representing between 0 and 200 ms of audio data.

F.4 Silence suppression

ITU-T Rec. H.225.0 states that coders may send silence frames before the stop transmission during a silence period. Since not all audio coders have in-band signalling for silence, a general mechanism at the RTP level should be defined. An example might be sending an empty RTP packet. This is for further study.

F.5 GSM codecs

GSM speech codecs include: GSM full rate (FR) [F-1], GSM half rate (HR) [F-2] and GSM enhanced full rate (EFR) [F-3]. Each codec produces three different speech traffic frame types, i.e.:

- Speech frames: Contains actual speech data;
- Idle frames: Indicates no voice activity, all data bits are set to one;
- Silence Descriptor (SID) frames: Indicates start of a silence period, data describes background noise. SID frames are marked inband with a fixed bit pattern.

F.5.1 Frame packetization

With all three GSM codecs speech traffic frame bits are packed into RTP frame most significant bit (MSB) first. One RTP packet may contain one or more GSM speech traffic frames. All endpoints shall be capable of receiving and identifying an idle frame. An idle GSM speech frame is filled with binary 1 s.

If an endpoint sets the `comfortNoise` parameter to `TRUE`, it shall send SID frames as specified in the comfort noise and discontinuous transmission (DTX) specifications of a particular GSM codec. During a silent period, a new SID frame, with (possibly) updated noise information, is sent periodically, that is every 24th frame. After a silence period, the marker bit shall be set to 1 in RTP header.

Full-rate codec

GSM full-rate codec sends a 260 bit (32.5 octets) frame every 20 ms. This information shall be packed into RTP frame with a four-bit prefix (0xD or 1101 binary), called signature. Therefore, GSM FR payload within RTP shall consist of 33 octets. SID (Silence Descriptor) frame is marked inband by a SID codeword stored into codec parameters as described in reference [F-4] below. The payload size of a SID frame is 33 octets. The signature of a full rate SID frame shall be same as that of a full rate speech frame (0xD). RTP coded FR speech shall have a bit rate of 13 200 bit/s, not including the packetization overhead.

Half-rate codec

GSM half-rate codec sends a 112 bit (14 octets)-frame every 20 ms. This information shall be packed into an RTP header without any prefixes/signatures. SID frame is marked inband by a SID codeword stored into codec parameters as described in reference [F-4] below. The payload size of a SID frame is 14 octets. RTP coded speech shall have a bit rate of 5600 bit/s, not including the packetization overhead.

Enhanced full rate

GSM EFR codec sends a 244 bit (30.5 octets)-frame every 20 ms. This information shall be packed into an RTP header with a four-bit prefix (0xC or 1100 binary), called "signature". Therefore, GSM EFR payload within RTP shall consist of 31 octets. SID frame is marked inband by a SID codeword stored into codec parameters as described in reference [F-4] below. The payload size of a SID frame is 31 octets. RTP-coded EFR speech shall have a bit rate of 12 400 bit/s, not including the packetization overhead.

F.5.2 Informative references

[F-1] GSM 06.10 (ETS 300 961), *Digital cellular telecommunications system; Full rate speech; Transcoding.*

[F-2] GSM 06.60 (ETS 300 726), *Digital cellular telecommunications system; Enhanced Full Rate (EFR) speech transcoding.*

[F-3] GSM 06.20 (ETS 300 969), *Digital cellular telecommunications system; Half rate speech; Half rate speech transcoding.*

[F-4] ETSI, TIPHON 03 001 (TS 101 318), *Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON); Using GSM speech codecs within ITU-T Recommendation H.323*.

[F-5] GSM 06.31 (ETS 300 963), *Digital cellular telecommunications system; Full rate speech; Comfort noise aspect for full rate speech traffic channels*.

[F-6] GSM 06.81 (ETS 300 729), *Digital cellular telecommunications system; Discontinuous Transmission (DTX) for Enhanced Full Rate (EFR) speech traffic channels*.

[F-7] GSM 06.41 (ETS 300 972), *Digital cellular telecommunications system; Half rate speech; Discontinuous Transmission (DTX) for half rate speech traffic channels*.

[F-8] GSM 06.12 (ETS 300 963), *Full rate speech; Comfort noise aspect for full rate speech traffic channels*.

[F-9] GSM 06.62 (ETS 300 728), *Digital cellular telecommunications system; Comfort noise aspects for Enhanced Full Rate (EFR) speech traffic channels*.

[F-10] GSM 06.22 (ETS 300 971), *Digital cellular telecommunications system; Half rate speech; Comfort noise aspect for the half Rate speech traffic channels*.

[F-11] GSM 08.60 (ETS 300 737), *Digital cellular telecommunications system; (Phase 2+) (GSM); In-band control of remote transcoders and rate adaptors for Enhanced Full Rate (EFR) and full rate traffic channels*.

F.6 G.722.1

The speech coding algorithm defined in ITU-T Rec. G.722.1 encodes wideband audio signals with a 50 Hz to 7 kHz bandwidth into one of two bit rates, 24 kbit/s or 32 kbit/s, using 20 ms frames and a sampling rate clock of 16 kHz. The bit rate can be changed at any 20-ms frame boundary, although rate change notification is not provided inband with the bitstream. When operating at 24 kbit/s, 480 bits (60 octets) are produced per frame, and when operating at 32 kbit/s, 640 bits (80 octets) are produced per frame. Thus, both bit rates allow for octet alignment without the need for padding bits.

The number of bits in a frame is fixed. However, within this fixed frame, G.722.1 uses variable length coding (e.g., Huffman coding) to represent most of the encoded parameters. Except for the categorization control bits parameter, all other bit stream parameters are represented by variable length codes, a variable number of bits. Figure F.6 illustrates this point and the order of the transmitted parameter fields. All variable length codes and the categorization control bits are transmitted in order from the leftmost (most significant (MSB)) bit to the rightmost (least significant (LSB)) bit. The use of Huffman coding means that it is not possible to identify the various coder parameters/fields contained within the bit stream without first completely decoding the entire frame.

Figure F.7 illustrates how the G.722.1 bit stream maps into an octet-aligned RTP payload. The encoder bit stream is split into a sequence of octets (60 or 80 depending on the bit rate), and each octet is in turn mapped into an RTP octet.

An RTP packet shall only contain G.722.1 frames of the same bit rate. The RTP timestamp shall be in units of 1/16 000th of a second.

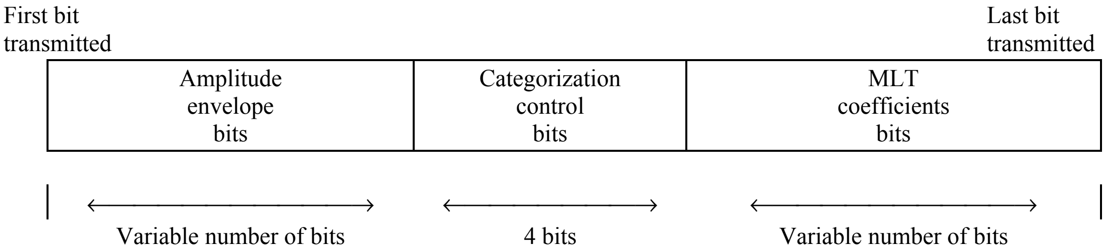


Figure F.6/H.225.0 – G.722.1 major bitstream fields and their order of transmission

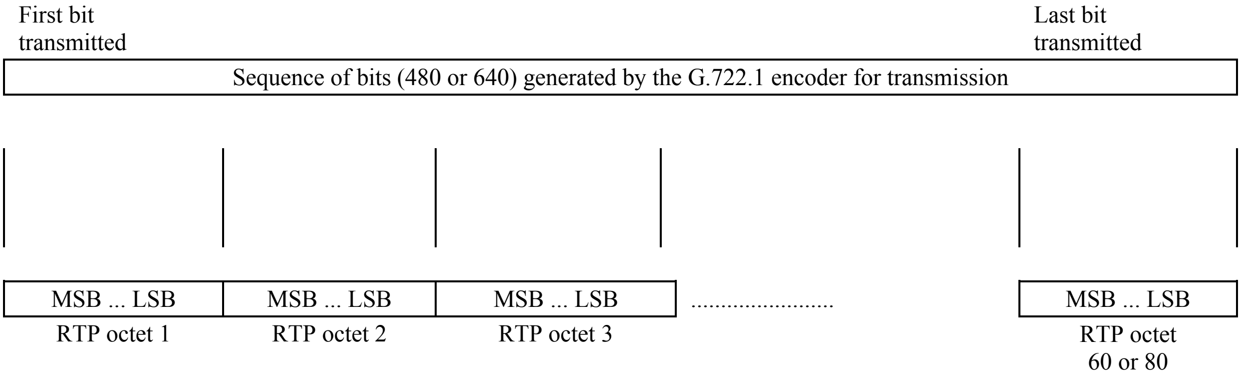


Figure F.7/H.225.0 – G.722.1 encoded bitstream mapping to RTP

F.7 TIA/EIA-136 ACELP

This vocoder is optimized for TIA/EIA-136 TDMA Digital Cellular and PCS systems. It includes voice activity detection (VAD), lost frame substitution and comfort noise generation (CNG) capabilities. The sampling rate is 8000 Hz and the compressed voice frame length is 20 ms. The vocoder produces a 148-bit speech-vector, s0 through s147, for each 20-ms voice frame. s0 is the most significant bit (MSB). Please refer to section 4 of reference [F7-1] for more details.

F.7.1 TIA/EIA-136 ACELP frame format

A speech indicator flag bit, SP, shall be generated by the vocoder and set to "1" to indicate a speech frame, or "0" to indicate a silence (comfort-noise) frame. This SP flag bit shall be inserted in bit-position 148. Bit position 149 is the BFI_CN (bad frame or comfort noise indicator) and bit position 150 is the CNU flag (comfort noise update). Bit position 151 shall always be set to 0.

The logical combinations of these three flags are described below.

The 152-bit (19-octet) transmit frame is depicted in Figure F.8. Octets are formed starting with the LSB and moving towards the MSB. The LSB is transmitted first.

bit 0 (MSB)	1 … 146	147	148	149	150	bit 151 (LSB)
s0	s1 … s146	S147	SP	BFI_CN	CNU	Always 0
Speech vector/Comfort noise			Flag	Flag	Flag	Padding bit

Figure F.8/H.225.0 – ACELP vocoder voice frame

F.7.2 TIA/EIA-136 ACELP silence suppression mode

In silence mode, the vocoder generates an ambient noise frame representation. This frame is used by the vocoder at the receiving end to regenerate the ambient noise of the transmitting end. The CN (comfort noise) parameters vector consists of only 38 bits, to which the three flag bits and seven padding bits (consisting of all zeros) are appended to form a six-octet frame.

The 48-bit (6-octet) CN frame is depicted in Figure F.9. Octets are formed starting with the LSB and moving towards the MSB. The LSB is transmitted first.

bit 0 (MSB)	**1 … 37**	**38**	**39**	**40**	**41**	**41-47 (LSB)**
Cn0	cn1 … cn37	S147	SP	BFI_CN	CNU	Always 0
Speech vector/Comfort noise			Flag	Flag	Flag	Padding bit

Key:

SP Speech indicator
BFI_CN Bad Frame Indicator/Comfort Noise Indicator
CNU Comfort Noise Update

The logical values of these flags and their meanings are defined below:

SP: 1 = speech frame; 0 = non-speech (comfort noise frame)

BFI_CN:
If SP = 1
And BFI_CN = 1
Then, this is a bad-voice frame
Otherwise (BFI_CN = 0), this is a good voice-frame

If SP = 0
And BFI_CN = 1
Then this is a bad comfort-noise frame
Otherwise (BFI_CN = 0), this is a good comfort-noise frame

CN:
If SP = 0
And BFI_CN = 0
And CN = 1
Then, this is an update comfort noise frame
Otherwise it is a non-valid CN frame

NOTE – A wireless mobile vocoder shall set the BFI_CN to 0. The receiving base station may set this flag to 1 if it is unable to correct errors introduced by the radio channel.

Figure F.9/H.225.0 – ACELP vocoder silence suppression frame

F.7.3 TIA/EIA-136 ACELP packetization

The packetization of IS-ACELP shall be in conformance with Annex B.

1) The packetization duration shall be a whole multiple of 20 ms.

2) A packet may consist of one or more frames each.

3) Codecs should be able to encode and decode several consecutive frames within a single packet.

4) All the bits of the encoded bit stream are transmitted always from the least significant bit towards the most significant bit.

F.7.4 TIA/EIA-136 ACELP referenced standard

[F7-1] TIA/EIA-136, part 410, *TDMA Cellular/PCS – Radio Interface, Enhanced Full Rate Voice Codec (ACELP)*. Formerly IS-641.

F.8 TIA/EIA-136 US1

This vocoder is optimized for TIA/EIA-136 TDMA Digital Cellular and PCS systems. Reference [F8-1] provides a detailed description of the vocoder.

F.8.1 TIA/EIA-136 US1 frame format

The sampling rate is 8000 Hz and the compressed voice frame-length is 20 ms. The vocoder produces 244 ordered bits per voice frame. Three flag bits, BFI, SID and TAF, are added to the speech vector. One padding bit (in bit position 247) is added to form a whole number of octets (31). The last bit is referred to as the least significant bit (LSB). This vocoder also supports DTX (discontinuous transmission) silence mode.

The transmit voice frame structure is shown in Figure F.10.

MSB – bit 0	1 … 243	244	245	246	247 (LSB)
s0	s1 … s243	BFI	SID	TAF	Always 0
Speech vector		Flag	Flag	Flag	Padding bit

Figure F.10/H.225.0 – US1 vocoder voice frame

F.8.2 TIA/EIA-136 US1 silence mode frames (TX-DTX)

In silence mode, special frames called SID (for silence descriptor) frames are transmitted in a schedule specified in section 1.3 of reference [F8-1].

A SID frame contains the same number of bits as normal speech frames, but the bit-map is different. See reference [F8-1] for details. The SID frame contains comfort noise (CN) parameters and a 95-bit SID code-word. The SID code-word is all "0"s. Other unused bits in the 244-bit vector payload are also set to "0". (See Figure F.11.)

MSB – bit 0	1 … 243	244	245	246	247 (LSB)
cn0	cn1 … cn243	BFI	SID	TAF	Always 0
Comfort noise vector		Flag	Flag	Flag	Padding bit

Figure F.11/H.225.0 – Base station to landline,
comfort noise transmit frame (US1)

The logic of the BFI, SID and TAF flags is similar to the equivalent flags of the TIA/EIA-136 ACELP vocoder, described in F.7.

F.8.3 TIA/EIA-136 US1 packetization

The packetization shall be in conformance with Annex B.

1) The packetization duration shall be a whole multiple of 20 ms.
2) A packet may consist of zero, one or more frames each.
3) Codecs should be able to encode and decode several consecutive frames within a single packet.
4) All the bits of the encoded bit stream are transmitted always from the least significant bit towards the most significant bit.

F.8.4 TIA/EIA-136 US1 reference standard

[F8-1] TIA/EIA-136, part 430, *TDMA Cellular/PCS – Radio Interface, US1 Full Rate Voice Codec*.

F.9 IS-127 EVRC

F.9.1 IS-127 EVRC description

F.9.1.1 General

The TIA/EIA IS-127 Enhanced Variable Rate Codec (EVRC) is optimized for TIA/EIA IS-95 CDMA Digital Cellular and PCS systems. The sampling rate is 8000 samples per second and the voice frame length is 20 ms (that is, 160 samples per frame). EVRC encodes active speech at full rate or half rate and background noise (no speech present) at one eighth rate. It delivers toll quality speech at very low average bit rate. A detailed description of the EVRC codec can be found in the publicly available TIA/EIA IS-127 standard [F9-1].

F.9.1.2 Compression rates

The EVRC coder compresses its input signal using one of three rates: full-rate (rate 1), half-rate (rate 1/2), and eighth-rate (rate 1/8). Full- and half-rates are used primarily for encoding active speech while the eighth-rate is used for encoding background noise (silence mode). All frames are 20 ms long, regardless of encoding rate.

F.9.1.3 Blanked packets

To allow for in-band signalling or for secondary traffic (see section 1.4.1 of [F9-1]), voice frames are blanked. The generated voice packet is simply not used and the decoder treats it as an erased packet. See [F9-1] for details.

F.9.1.4 Half rate

Half-rate encoding is used, instead of the normal full-rate, when a signalling message has to be added to the traffic channel.

F.9.1.5 Null 1/8 rate traffic channel data

A rate one-eighth packet in which all bits are set to "1" is considered null Traffic Channel data. Such packets are declared "erased packets" and are handled as described in section 5 of [F9-1].

Rate information and channel coding bits are added to the vocoder output bits for transport over the air, in accordance with TIA/EIA IS-95.

The packet types, number of bits-per-packet, raw vocoder bit-rates and the aggregate rates (vocoder bits plus additional bits) are shown in Table F.3.

Table F.3/H.225.0 – EVRC packets and bit rates

Packet type (3 bits)	Rate	Bits/packet	Vocoder bit rate kbit/s	Aggregate rate kbit/s
1	Full	171	8.55	9.6
2	Half	80	4.0	4.8
3 (Note)	Fourth (service option-1 compatibility)	40		
4	Eighth	16	0.8	1.2
5	Blanked	0	–	–
6	Full-rate with errors	171	–	–
7	Bad frame (erasure)	0	–	–
NOTE – Type 3 packets may only be generated by older IS-96 encoders. The IS-127 decoder shall treat these packets as erased-packets.				

F.9.2 IS-127 EVRC packetization

F.9.2.1 General requirements

The transmission packetization shall be in conformance with Annex B.

1) The packetization duration shall be a whole multiple of 20 ms.

2) A transmission packet may consist of zero, one, or more frames.

3) Codecs should be able to encode and decode several consecutive frames within a single transmission packet.

4) All the bits of the encoded bit stream shall always be transmitted from the least significant bit towards the most significant bit.

F.9.2.2 Frame formats

F.9.2.2.1 Full rate – F1

The EVRC full-rate, 176-bit (22-octet) transmit frame (F1), is depicted in Figure F.12. Octets are formed starting with the LSB and moving towards the MSB. The LSB (bit 175) is transmitted first.

Bit 0 (MSB)	Bits 1 through 170	Bits 171 through 175 (LSB)
s0	s1 … s170	Always 0
Speech vector		Padding bits

Figure F.12/H.225.0 – F1, Full-rate EVRC frame

F.9.2.2.2 Half rate – F2

The EVRC half-rate, 80-bit (10-octet) transmit frame (F2) is depicted in Figure F.13. Octets are formed starting with the LSB and moving towards the MSB. The LSB (bit 79) is transmitted first.

Bit 0 (MSB)	Bits 1 through 79 (LSB)
s0	s1 … s79
Speech vector	

Figure F.13/H.225.0 – F2, Half-rate EVRC frame

F.9.2.2.3 Eighth Rate – F3

The EVRC eighth-rate, 16-bit (2-octets) transmit frame (F3) is depicted in Figure F.14. Octets are formed starting with the LSB and moving towards the MSB. The LSB (bit 15) is transmitted first.

Bit 0 (MSB)	Bits 1 through 15 (LSB)
s0	s1 … s15
Speech vector	

Figure F.14/H.225.0 – F3, Eighth-rate EVRC frame

F.9.3 IS-127 EVRC reference standards

[F9-1] TIA/EIA IS-127 (1997), *Enhanced Variable Rate Codec, Speech Service Option 3 for Wideband Spread Spectrum Digital Systems*.

[F9-2] TIA/EIA IS-95-B (1999), *Mobile Station-Base Station Compatibility Standard for Wideband Spread Spectrum Cellular Systems*.

F.10 H.223 MUX-PDU packetization

F.10.1 Introduction

The H.223 MUX-PDU is used by a packet-oriented multiplexing protocol designed for the exchange of one or more information streams between higher-layer entities such as data and control protocols and audio and video codecs, as defined in ITU-T Rec. H.223.

Each information stream is represented by an H.245 unidirectional logical channel which is identified by a unique Logical Channel Number (LCN), an integer between 0 and 65535. LCN 0 is a permanent logical channel assigned to the H.245 control channel. All other logical channels are dynamically opened and closed by the transmitter using the H.245 OpenLogicalChannel and CloseLogicalChannel messages. All necessary attributes of the logical channel are specified in the OpenLogicalChannel message. For applications that require a reverse channel, a procedure for opening bidirectional logical channels is also defined in ITU-T Rec. H.245.

The general structure of the multiplexer is shown in Figure 2/H.223. The multiplexer consists of two distinct layers: a Multiplex (MUX) layer and an Adaptation Layer (AL).

Support of the H.223 payload type is signalled using H.245 capability sets and in the H.245 openLogicalChannel message using RTP dynamic payload types.

F.10.2 MUX-PDU packetization format

The H.223 MUX-PDU specified by Figure 3/H.223 is carried as payload data within the RTP protocol. The order of bit transmission is specified in 3.2.2/H.223, and the field mapping convention is in 3.2.3/H.223.

Though a MUX-PDU can occupy more than one RTP packet, a MUX-PDU shall start with the first octet of an RTP packet payload.

Each RTP packet contains a timestamp that is derived from the sender's clock reference. The timestamp shall represent the target transmission time of the first byte of the H.223 MUX-PDU. The primary purpose of this timestamp is for the receiver to estimate and reduce any network-induced jitter, and to reproduce the H.223 bitstream with constant bit rate.

The usage of the fields of the RTP header shall be as follows:

1) An RTP dynamic payload type is used.

2) The RTP timestamp represents the target transmission time for the first byte of the MUX-PDU in the packet over the H.223 constant bitrate channel. This timestamp is derived

from the clock frequency with a default value of 90 kHz. The sender can change this frequency, and the selected value is signalled by the **BitRate** parameter in the **H223Capability** structure in H.245 messages. If a MUX-PDU occupies more than one RTP packet, the RTP timestamp shall be the same on successive packets. The timestamp should be calculated based on the number of bytes included in the transmitted MUX-PDUs.

3) The marker bit of the RTP header is set to one in the last packet of a MUX-PDU, and otherwise must be zero. Thus, it is not necessary to wait for a following packet to detect the MUX-PDU boundary.

The H.223 MUX-PDU follows the RTP header, as in:

RTP header	MUX-PDU data

Annex G

Communication between and within administrative domains

G.1 Scope

It is expected that the overall H.323 network will consist of smaller subsets of equipment organized in some manner, such as by Administrative Domain. Because of the potentially large numbers of H.323 elements that will exist in H.323 networks, an efficient protocol is needed to allow calls to be completed between Administrative Domains. The most elementary example is for a user (an endpoint) in one Administrative Domain to reach a user (an endpoint) serviced by another Administrative Domain. While the H.225.0 RAS protocol can address many of the needs for communication between Administrative Domains, it is neither complete nor efficient for this purpose.

For the same reason, an efficient protocol also needs to be specified between H.323 elements within the same Administrative Domain.

This annex describes methods to allow address resolution, access authorization and usage reporting between and within Administrative Domains in H.323 systems for the purpose of completing calls. H.323 elements that communicate using the procedures described in this annex are known as Peer Elements. An Administrative Domain exposes itself to other Administrative Domains through a type of logical element known as a Border Element. Border Elements are special cases of Peer Elements, at least one of whose peers belongs to another Administrative Domain. A Peer Element may be colocated with any other entity (for example, with a gatekeeper). Annex G does not require an Administrative Domain to reveal details about its organization or architecture. Annex G does not mandate a specific system architecture within an Administrative Domain. Furthermore, Annex G supports the use of any call model (gatekeeper routed versus direct endpoint).

The general procedure is for Peer Elements to exchange information regarding the addresses each can resolve. Border Elements exchange information regarding the addresses their Administrative Domains can resolve. Addresses can be specified in a general manner or in an increasingly specific manner. Additional information allows elements within an Administrative Domain to determine the most appropriate Administrative Domain to serve as the destination for the call. Border Elements may control access to their exposed addresses, and require reports on the usage made during calls to those addresses.

Figure G.1 indicates a number of reference points representing signalling among various elements in an H.323 network. In Figure G.1, the Administrative Domains are part of a global packet network without edges. Note that Figure G.1 is not an explicit definition of an H.323 system architecture, but is meant to illustrate signalling reference points.

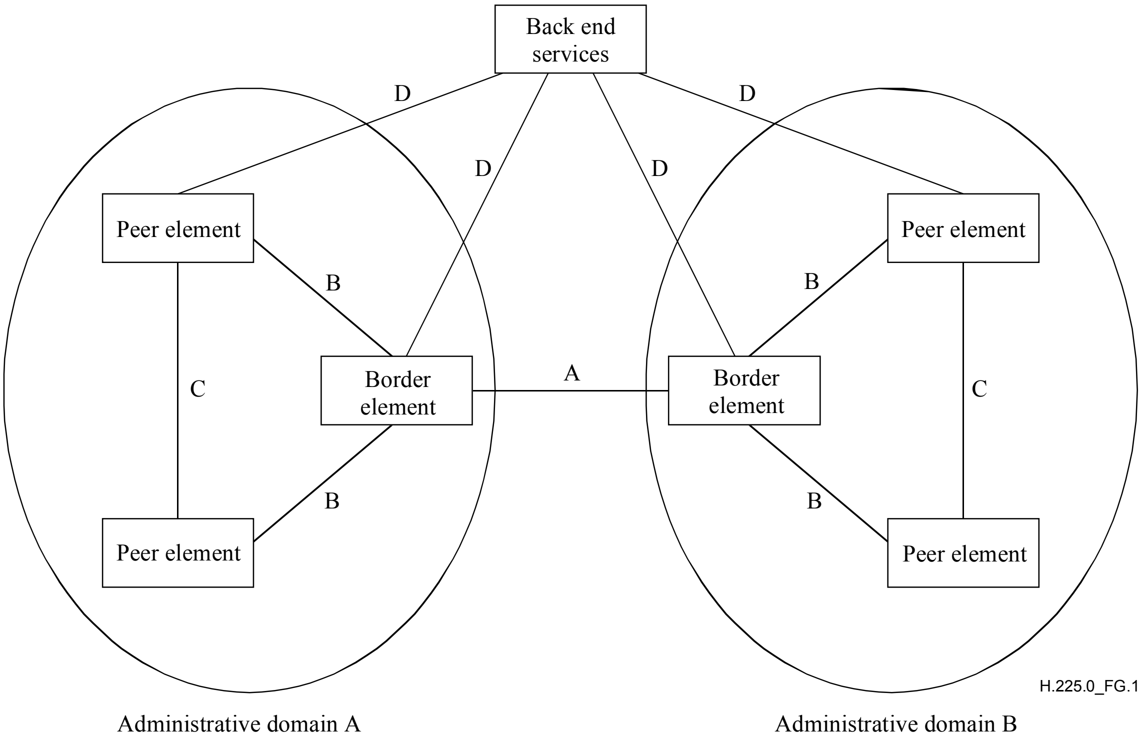


Figure G.1/H.225.0 – System reference points

The figure indicates the following reference points:

A – Between Border Elements belonging to different Administrative Domains.

B – Between Border Elements and Peer Elements within the same Domain.

C – Between Peer Elements within the same Domain.

D – Between H.323 elements and back-end services (not in the scope of this annex).

Reference points A, B and C are the focus of Annex G. As was mentioned earlier, a Peer Element may be colocated with some other H.323 element.

Clause G.7, Signalling Examples, provides some signalling examples that may aid understanding.

G.2 Definitions

This annex defines the following terms:

G.2.1 administrative domain: An Administrative Domain is a collection of H.323 entities administered by one administrative entity. An Administrative Domain can consist of one or more gatekeepers (that is, one or more zones).

G.2.2 back-end services: Back-End Services are functions such as user authentication or authorization, accounting, billing, rating/tariffing, etc., Back-end services and the protocol to exchange information with back-end services (if different than that in this annex) are not in the scope of this annex.

G.2.3 peer element: As defined in ITU-T Rec. H.501, a Peer Element is a logical element that originates or terminates signalling messages defined in that Recommendation. This element may exist in combination with other H.323 elements, for example a combination of Peer Element, gatekeeper and gateway. An Administrative Domain may contain any number of Peer Elements.

G.2.4 border element: A special case of the Peer Element, the Border Element is a functional element with at least one peer that is outside of its Administrative Domain. It supports public access into an Administrative Domain for the purposes of call completion or any other services that involve multimedia communication with other elements within the Administrative Domain. The Border Element controls the external view of the Administrative Domain.

G.2.5 clearing house: A service (possibly in the form of a Border Element) that can provide resolution for all addresses (i.e., a type of aggregation point).

G.3 Abbreviations

This annex uses the following abbreviations:

AD	Administrative Domain
BE	Border Element
CH	Clearing House
DST	Daylight Saving Time
EP	Endpoint
GK	Gatekeeper
GW	Gateway
PE	Peer Element
SCN	Switched Circuit Network
T	Terminal

G.4 Normative 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.

[1] ITU-T Recommendation H.225.0 Version 4 (2000), *Call signalling protocols and media stream packetization for packet-based multimedia communication systems*.

[2] ITU-T Recommendation H.235 Version 2 (2000), *Security and encryption for H-series (H.323 and other H.245-based multimedia terminals)*.

[3] ITU-T Recommendation H.323 Version 4 (2000), *Packet-based multimedia communications systems*.

[4] ITU-T Recommendation H.323 (2000) Annex K, *HTTP-based Service Control Transport Channel in H.323*.

[5] ITU-T Recommendation H.501 (2002), *Protocol for mobility management and intra/inter-domain communication in multimedia systems*.

[6] ITU-T Recommendation H.460.2 (2001), *Number Portability Interworking between H.323 and SCN networks*.

G.5 System models

Annex G does not mandate a specific system architecture among Administrative Domains or within an Administrative Domain. The following subclauses provide some sample architectures, but these are to be viewed as illustrative rather than exhaustive.

Remember that a Peer Element is a functional element that may exist together with any other H.323 element. Figure G.2 shows some examples of Peer Element implementations in combination with other elements.

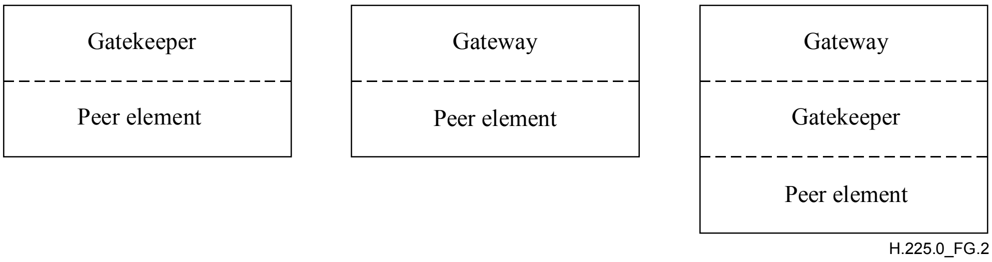


Figure G.2/H.225.0 – Peer element placement examples

In general, an Administrative Domain is viewed as consisting of any number of zones and any number of Peer Elements. The relationships among Administrative Domains, and among Peer Elements within an Administrative Domain, may be any of a variety of organizations. The following clauses describe example relationships and organizations. These are described as between Administrative Domains, but the Hierarchical, Distributed/Full-Mesh and Aggregation examples could also be used to organize Peer Elements within an Administrative Domain.

Note again that the following examples are illustrative, and not meant to exclude other possible organizations.

G.5.1 Hierarchical

Figure G.3 shows a simple hierarchical arrangement among Administrative Domains. In such an arrangement, a Border Element in an Administrative Domain would consult a Border Element in an Administrative Domain higher in the hierarchy to resolve an address.

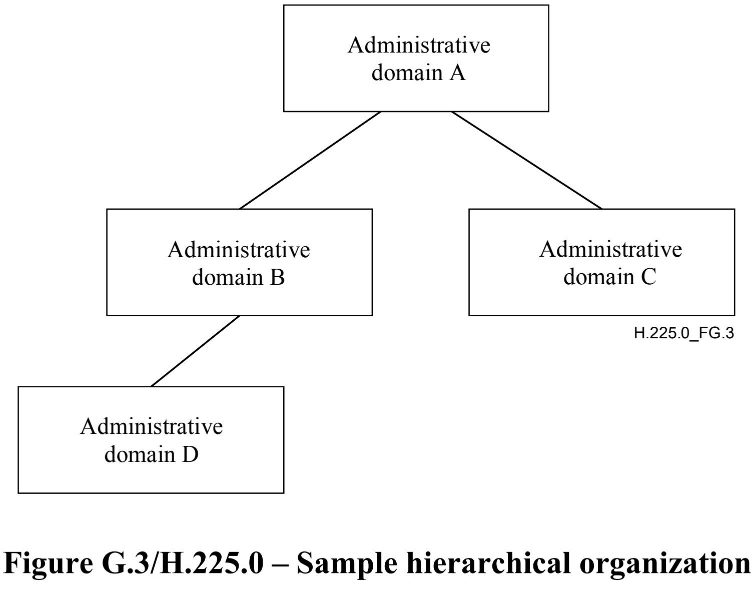


Figure G.3/H.225.0 – Sample hierarchical organization

G.5.2 Distributed or full mesh

An entirely distributed or full mesh model is possible, as shown in Figure G.4. In this example, a Border Element in each Administrative Domain communicates with Border Elements in the other known Administrative Domains.

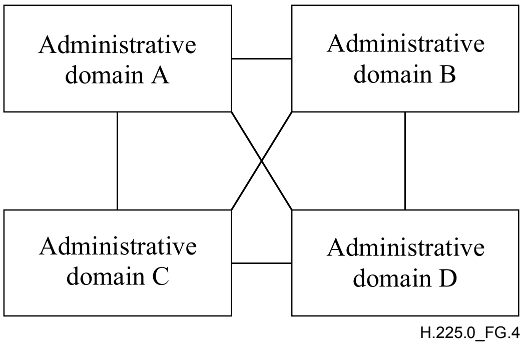


Figure G.4/H.225.0 – Sample distributed organization

G.5.3 Clearing house

An example of a Clearing House arrangement is shown in Figure G.5. In this arrangement, each Administrative Domain consults the Clearing House to resolve addresses. Note that since a Clearing House is an entity that exists outside of an Administrative Domain, the Peer Elements that communicate with it are by definition Border Elements.

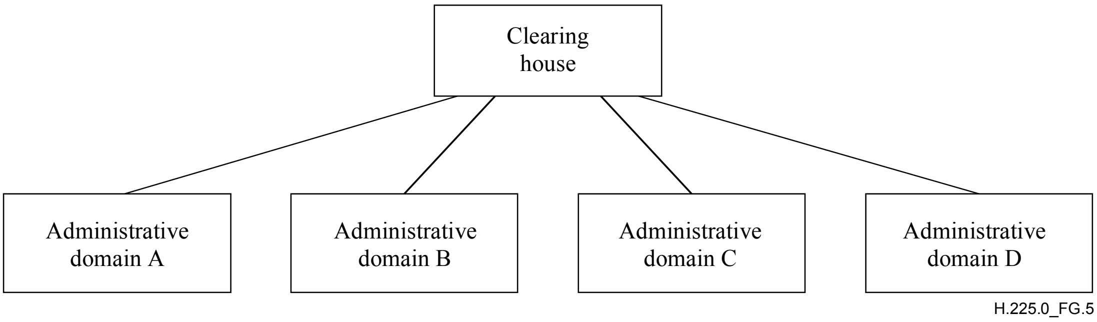


Figure G.5/H.225.0 – Sample clearing house organization

G.5.4 Aggregation point

Figure G.6 shows an example of an aggregation point. In this example, Administrative Domain B is an aggregation point that can provide address resolution for both itself and Administrative Domains C and D. As an example, Administrative Domain B may forward resolution requests from Administrative Domain A to Administrative Domain C, or may instruct Administrative Domain A to contact Administrative Domain C directly for certain destinations. If Administrative Domain B forwards a request from Administrative Domain A to Administrative Domain C, Administrative Domain B may cache Administrative Domain C's response.

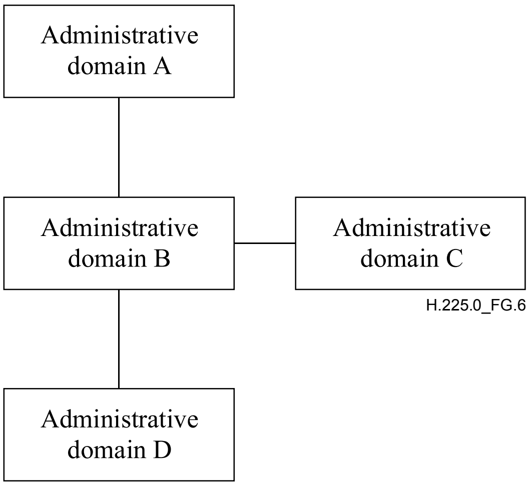


Figure G.6/H.225.0 – Aggregation point example

G.5.5 Overlapping administrative domains

More than one Administrative Domain may be able to resolve a given address. For example, multiple Administrative Domains could contain gateways that can complete a call to a terminal in the GSTN. The selection of the appropriate destination Administrative Domain is the responsibility of the Administrative Domain of origin. The algorithm employed to select the destination Administrative Domain is an implementation matter.

G.6 Operation

G.6.1 Use of H.501 messages

Annex G/H.225.0 implementations shall make use of messages defined in ITU-T Rec. H.501. The entities exchanging H.501 messages are referred to in that Recommendation as Peer Elements.

The following is a list of the H.501 messages used by Annex G/H.225.0:

ServiceRequest

ServiceConfirmation

ServiceRejection

ServiceRelease

DescriptorRequest

DescriptorConfirmation

DescriptorRejection

DescriptorIDRequest

DescriptorIDConfirmation

DescriptorIDRejection

DescriptorUpdate

DescriptorUpdateAck

AccessRequest

AccessConfirmation

AccessRejection

RequestInProgress

NonStandardRequest

NonStandardConfirmation

NonStandardRejection

UnknownMessageResponse

UsageRequest

UsageConfirmation

UsageRejection

UsageIndication

UsageIndicationConfirmation

UsageIndicationRejection

ValidationRequest

ValidationConfirmation

ValidationRejection

An Annex G/H.225.0 Peer Element receiving an H.501 request message not included in the above list shall respond with an UnknownMessageResponse message.

Messages shall contain all fields defined by ITU-T Rec. H.501 as mandatory, and may contain optional fields as required.

G.6.2 Address templates and descriptors

A Peer Element obtains templates in these ways:

- static configuration;
- receiving descriptors from other Peer Elements in response to general requests;
- receiving responses to specific queries.

G.6.2.1 Static configuration

A Peer Element will maintain templates for all the zones for which it is responsible. These templates may be explicitly provisioned in the Peer Element, or, in the case where the Peer Element co-exists with gatekeepers, these templates may be formed by summarizing information obtained from each gatekeeper with which the Peer Element communicates. The Peer Element may make this information available to other Peer Elements via responses to requests. An Administrative Domain may choose the level of detail to be provided by its Border Element(s). Examples include:

- A Border Element that wishes to hide internal structure might provide one descriptor (with an indication to send an AccessRequest message) that describes its whole zone and refers to a gatekeeper that will handle all incoming calls.
- A Border Element that does not care about revealing internal structure might provide a set of templates, each describing the gatekeeper for a zone within the domain.
- A Border Element that is on a firewall (or one using the gatekeeper routed model) might provide a template for the whole zone with an indication to send a Setup message.
- A Border Element with holes in its domain (because numbers have been moved to another Administrative Domain) provides templates marked **sendAccessRequest** that indicate which Border Element should be used to contact the other Administrative Domain.
- A Clearing House Border Element (such as one that has a complete copy of, e.g., 44) might hold a template marked **sendAccessRequest** for each Administrative Domain within 44.

Peer Elements need not keep a copy of the whole database. If a Peer Element does not hold a copy of the whole database, then it should contain statically configured **sendAccessRequest** templates indicating a Clearing House Border Element that will be used to resolve other queries.

G.6.2.2 Receiving descriptors

A Peer Element may request the statically configured templates from another Peer Element. The response to the request is decided by the Peer Element from which the templates are being requested. To request a transfer, the Peer Element sends a DescriptorRequest message specifying the descriptors it wishes to receive. If the owning Peer Element is able to transfer the descriptors, it responds with a DescriptorConfirmation message specifying all the templates.

The requesting Peer Element may cache a copy of a template received in this manner until the template's lifetime expires, at which point the Peer Element should delete its copy of the template. If the owning Peer Element changes its statically configured templates before their lifetime has expired, then it shall send a DescriptorUpdate message to those Peer Elements of which it is aware. A Peer Element in receipt of a DescriptorUpdate message should delete, add, or change all indicated templates in its cache, or should request copies of the indicated descriptors from the owner.

An intermediate Peer Element (a Peer Element between the originating and destination Administrative Domains, such as a Clearing House or aggregation point) may publish its own descriptors based on the descriptors it receives. For example, a Clearing House may indicate itself as the contact for an AccessRequest message even though the descriptors it received from another Border Element indicate that other Border Element as the contact.

A Peer Element may indicate in a template the requirement for an originator to receive permission to place a call into an Administrative Domain. When the **callSpecific** flag is set in a template and the message type indicates that an AccessRequest message shall be sent, the originator shall provide per-call information in the AccessRequest message. If a Peer Element receives the AccessRequest message without per-call information and policy is to require per-call information, the Peer Element shall reply with an AccessRejection message with a reason of **needCallInformation**.

A Peer Element may send a DescriptorUpdate message to other known Peer Elements, or the Peer Element may multicast a DescriptorUpdate message. If a DescriptorUpdate message is multicast, the Peer Element should consider the scope of the multicast. The DescriptorUpdate message may contain the descriptors that have changed. Alternatively, the DescriptorUpdate message may indicate only the identification of the descriptors that changed, allowing the recipient to query for the new information. If a large number of descriptors have changed, the information should be sent in multiple DescriptorUpdate messages so that a particular DescriptorUpdate message does not exceed the maximum transport packet size.

G.6.2.3 Receiving responses to specific queries

A Peer Element may send an AccessRequest message to another Peer Element asking for the resolution of a fully qualified or partially qualified address. The AccessRequest is usually sent over unreliable transport (e.g., UDP), although it may be sent over reliable transport (e.g., TCP).

A Peer Element in receipt of an AccessRequest searches its database and responds with the most specific template for the destination. If multiple templates satisfy the request then the Peer Element shall return all matching templates. If the destination Peer Element is actually responsible for the alias address specified, the Peer Element will usually respond with a template indicating that either an AccessRequest or Setup message should be sent. If the destination Peer Element is a Clearing House, it will normally respond with a template indicating that the AccessRequest message should be sent.

The destination Peer Element may also add templates to the response that it believes will be useful in the future. The addition of these templates should not make the response so large that the transport network will need to fragment it (e.g., 576 octets for IPv4 or 1200 octets for IPv6).

For example, a Border Element that is tightly coupled with a firewall may provide two templates in its response to AccessRequest messages: one template with a short lifetime (of a few minutes or seconds) specifying the location to which a Setup message should be sent, and additional templates specifying that AccessRequest messages should be sent to the Border Element for other AliasAddresses within the Administrative Domain.

A Peer Element may cache a template received in an AccessConfirmation until its lifetime expires.

G.6.3 Discovery of a peer element or set of peer elements

G.6.3.1 Static

A Peer Element may have an administered set of other Peer Elements that it may contact for address resolution. This administered set may be defined through a set of bilateral agreements, e.g., between an Administrative Domains and other Administrative Domains. The Administrative Domains may optionally utilize the service of a Clearing House.

G.6.3.2 Dynamic

On IP networks, ownership of Email-ID style addresses is defined by the DNS system. Thus, in the absence of any better information, a border element may do a DNS SRV record lookup on the part of the email-ID to the right of the "@" sign (for example, a DNS SRV lookup on **_h2250-annex-g._udp.example.org** for **person@example.org**). The response to this lookup should be used to synthesize a **sendAccessRequest** template that can be used during the resolution process. Templates synthesized from DNS requests should not be cached for longer than the lifetime provided in the DNS response.

G.6.3.3 Other methods

The use of other methods to locate another Peer Element are for further study.

G.6.4 Resolution procedures

G.6.4.1 Resolution procedure within administrative domain

When a Peer Element is asked to resolve an AliasAddress (e.g., by a colocated gateway or gatekeeper), it finds matching templates in its cache.

If more than one template matches, appropriate templates are selected and sorted according to local policy. For example, templates may be first sorted by wildcard length (more specific templates are better), then sorted by the type of protocol specified (**sendSetup** is better than **sendAccessRequest**).

If multiple templates satisfy the request, then the Peer Element shall return all matching templates.

If the template selection procedure produces no templates marked as **sendSetup**, then the Peer Element sends an AccessRequest message with a specific destination address to the address specified in the template. When it gets an answer from the Peer Element, it may store that in its cache and return to the requester the address to which to send the Setup message.

G.6.4.2 Resolution procedure between administrative domains

When a Border Element receives an AccessRequest message from a Border Element in another Administrative Domain, it searches through the templates in its cache and finds those that match the address in the query.

If more than one template matches, the matching templates are first sorted by wildcard length (more specific templates are better). They are then sorted by the specified message type (**sendSetup** is better than **sendAccessRequest**). In each case all templates other than the most specific match are discarded.

If the matched templates are marked as **sendAccessRequest** then the Border Element may choose to forward the AccessRequest message to the Border Element(s) specified in the template(s), or may choose to return the templates as they are. If the hop counter in the received AccessRequest message has reached zero, then the Border Element cannot forward the AccessRequest message to another Peer Element, but should instead return any matching templates. If the hop counter has reached zero and the Border Element has no information to provide in an AccessConfirmation, the Border Element should respond with an AccessRejection message indicating that the hop count was exceeded.

At this point, the Border Element may use another Border Element (e.g., a Clearing House) to authorize the access request. To do that, it sends a ValidationRequest message, carrying access tokens supplied by the requesting Border Element in the AccessRequest rights. The recipient Border Element validates the tokens and returns ValidationConfirmation.

The Border Element then returns an AccessConfirmation message containing the templates that it has found (these will have the same address and message type fields) and any other templates that it considers useful.

If multiple templates satisfy the request, then the Border Element shall return all matching templates.

If the access request contains specific call information, then the returned templates are valid only for the call requested. This is used when an Administrative Domain wishes to grant access on a per-call basis. In that case, the Administrative Domain may mandate the inclusion of call information per each AccessRequest sent to it. It does so by setting a flag in the templates that refer to it.

G.6.5 Usage information exchange

Peer Elements may request other Peer Elements to provide them information about the usage of resources in specific calls. UsageIndication messages may be provided at any stage of the call. Also, multiple UsageIndication messages may be sent for the same call, each one with possibly more up-to-date information, or reporting on consecutive call segments or different media type usage. See G.6.5.1 for details.

UsageIndication messages may be exchanged irrespective of whether the two Peer Elements have a service relationship between them. However the policy of a Peer Element may not allow such exchanges without a service relationship. In such a case, the Peer Element may reject the UsageIndication message, with a reject code of `noServiceRelationship`.

UsageIndication requests shall be sent whenever a Peer Element requires them, either in the templates for which it serves as contact, or by indicating in the ServiceRequest message it sends during service relationship establishment with a remote Peer Element, or by so indicating in either of the UsageRequest, AccessRequest, ValidationRequest and ValidationConfirmation messages sent in the context of the call for which usage information is required.

G.6.5.1 Multiple UsageIndications for the same call

Multiple UsageIndications for the same call provide increasingly more up-to-date information on the same media types, or usage information about new media types created in the same call. Also, since Peer Elements may take over calls while being in progress, not all the UsageIndications necessarily originate from the same Peer Element. The following rules define the semantics:

1) A UsageIndication message received with a `usageCallStatus` of `callInProgress` implies a subsequent UsageIndication with the same `callIdentifier` and `senderRole` should be received. If the recipient is configured for fault recovery, it may choose to conclude after a configured time interval with no further UsageIndication messages that a fault has occurred, and may recover whatever data it can from the received UsageIndication messages.

2) Subsequent UsageIndication messages with the same `usageField` ids should report a `startTime` matching the `endTime` of the previous message (although this may be impossible for an alternate Peer Element). Recipients shall assume each report is for a distinct period.

Other information in the `usageField` overrides the information received in previous messages with the same `usageField` id.

3) A Peer Element should send a new UsageIndication message for each change in the media type during the call, e.g., audio stopped and fax started, or a codec has changed. If multiple media types are engaged at the same time (e.g., audio and video) they should be reported in the same UsageIndication message.

G.6.5.2 Requesting and negotiating usage information during service relationship establishment

A Peer Element PE_A may include a `UsageSpecification` element in a ServiceRequest message to a second Peer Element PE_B. This `UsageSpecification` element will be used to define the default usage information to be reported for all calls that take place while the service relationship exists between the two Peer Elements PE_A and PE_B. This `UsageSpecification` shall be used for all calls for which PE_B sends UsageIndications to PE_A.

If a `UsageSpecification` element arrives at PE_B in another message from PE_A (for example, an AccessConfirmation), then the new `UsageSpecification` overrides the default `UsageSpecification` for all calls related to the new message.

A Peer Element receiving a ServiceRequest that contains a `UsageSpecification` element should act as follows:

i) If the receiving Peer Element is willing to accept the ServiceRequest and the `UsageSpecification` contained within, it shall send a ServiceConfirmation message that contains the same `UsageSpecification` as the one received in the ServiceRequest. The `UsageSpecification` shall apply to both incoming calls to the recipient Peer Element from the requesting Peer Element and outgoing calls from the recipient Peer Element to the requesting Peer Element.

ii) If the receiving Peer Element is willing to accept the ServiceRequest but is not willing to accept the `UsageSpecification` contained within, it shall either send a ServiceConfirmation message containing a different `UsageSpecification` that specifies the usage information that it is able to provide to the requesting Peer Element, or a ServiceRejection message with the reason set to **cannotSupportUsageSpec**.

iii) If the receiving Peer Element does not support usage reporting at all, it shall return a ServiceRejection message with the reason set to `usageUnavailable`.

A Peer Element receiving a ServiceConfirmation should act as follows:

i) If the `UsageSpecification` in the ServiceConfirmation is the same as the one sent in the ServiceRequest, then the originating Peer Element and terminating Peer Element have established a service relationship between them.

ii) If the `UsageSpecification` in the ServiceConfirmation is different than the one sent in the ServiceRequest message, then if the originating Peer Element is willing to use the new `UsageSpecification`, the service relationship is established. If the originating Peer Element is not willing to use the new `UsageSpecification`, it shall send a ServiceRelease message with reason set to `terminated`. The originating Peer Element could then analyze the `UsageSpecification` returned in the ServiceConfirmation in order to build a new ServiceRequest message with a modified `UsageSpecification` that may be acceptable to both Peer Elements.

iii) If the ServiceConfirmation does not contain a `UsageSpecification` (and the ServiceRequest did), then the Peer Element that sent the ServiceConfirmation cannot or will not employ usage reporting at the level of the service relationship. This is the case when, for example, the recipient Peer Element implements version 1 of this annex. In this case, the originating Peer Element can either terminate the service relationship (by sending a

ServiceRelease message with the reason code set to **terminated**), or not terminate the service relationship. In either case, if the originating Peer Element is interested in receiving usage information about calls, it should request usage information using the mechanisms described in version 1 of this annex (i.e., sending **UsageSpecification** elements in either AccessRequest, AccessConfirmation (within the returned address templates), UsageRequest, ValidationRequest or ValidationConfirmation messages).

G.6.6 Number portability information signalling

ITU-T Rec. H.460.2 describes mechanisms for number portability in H.323 networks. Support for H.460.2 requires that Annex G be capable of transporting number portability information through address resolution message exchanges. The interface between the Annex G Border Element and the other H.323 network elements with which it communicates is not covered by this annex; it is assumed this interface is capable of transporting the H.460.2 number portability to and from the Annex G Border Element.

When an AccessRequest is sent, it will transport the H.460.2 number portability information, if present, using the **genericData** field in the common information portion of the message.

AccessConfirmation and AccessRejection messages will carry the corresponding number portability response information in the **genericData** field as well. In the case of an AccessRejection, the reject reason shall be **genericDataReason**.

G.7 Signalling examples

These signalling examples are provided to illustrate basic operation. In these examples, assume that the Administrative Domains have agreements with each other, so the Border Elements have been provisioned with information (e.g., TCP ports) about each other. In many of the examples below, RAS LRQ/LCF messages are exchanged between a gatekeeper and a Border Element within the same Administrative Domain. This is purely for illustrative purposes, and analogous Annex G messages could be exchanged between the Border Element and a Peer Element residing within the gatekeeper.

G.7.1 Distributed or full mesh

An example of a distributed network is shown in Figure G.7.

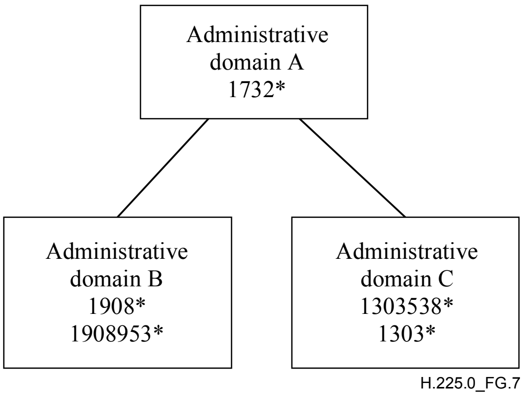


Figure G.7/H.225.0 – Distributed network for signalling examples

For this example, assume the Administrative Domains each have one Border Element, and that the Border Elements are configured to resolve addresses as follows:

Administrative domain	Template definition	Comment
A	Descriptor "d1": Pattern = 1732* Transport address = BE_A call signal address Message type = **sendSetup**	Signalling for any call into AD A will be through AD A's Border Element.
B	Descriptor "d2": Pattern = 1908* Transport address = BE_B Annex G address Message type = **sendAccessRequest**	For calls to 1908*, an AccessRequest message is needed to get the destination's (i.e., a gateway) call signalling address.
	Descriptor "d3": Pattern = 1908953* Transport address = GW_{B1} call signalling address Message type = **sendSetup**	For calls to 1908953*, the Setup can be sent directly to this particular gateway.
C	Descriptor "d4": Pattern = 1303538* Transport address = GK_{C1} call signal address Message type = **sendSetup**	Calls to 1303538* will be routed through this particular gatekeeper.
	Descriptor "d5": Pattern = 1303* Transport address = BE_C Annex G address Message type = **sendAccessRequest**	Calls to 1303* can be signaled directly to the destination gateway, but an AccessRequest must be sent to obtain the gateway's call signalling address.

G.7.1.1 Exchange of zone information

In the distributed, or full mesh, organization each Administrative Domain is aware of each other Administrative Domain, presumably through a number of bilateral contractual agreements. At any time, a Border Element in an Administrative Domain can query another Administrative Domain to obtain addressing information. An example of this signalling appears in Figure G.8.

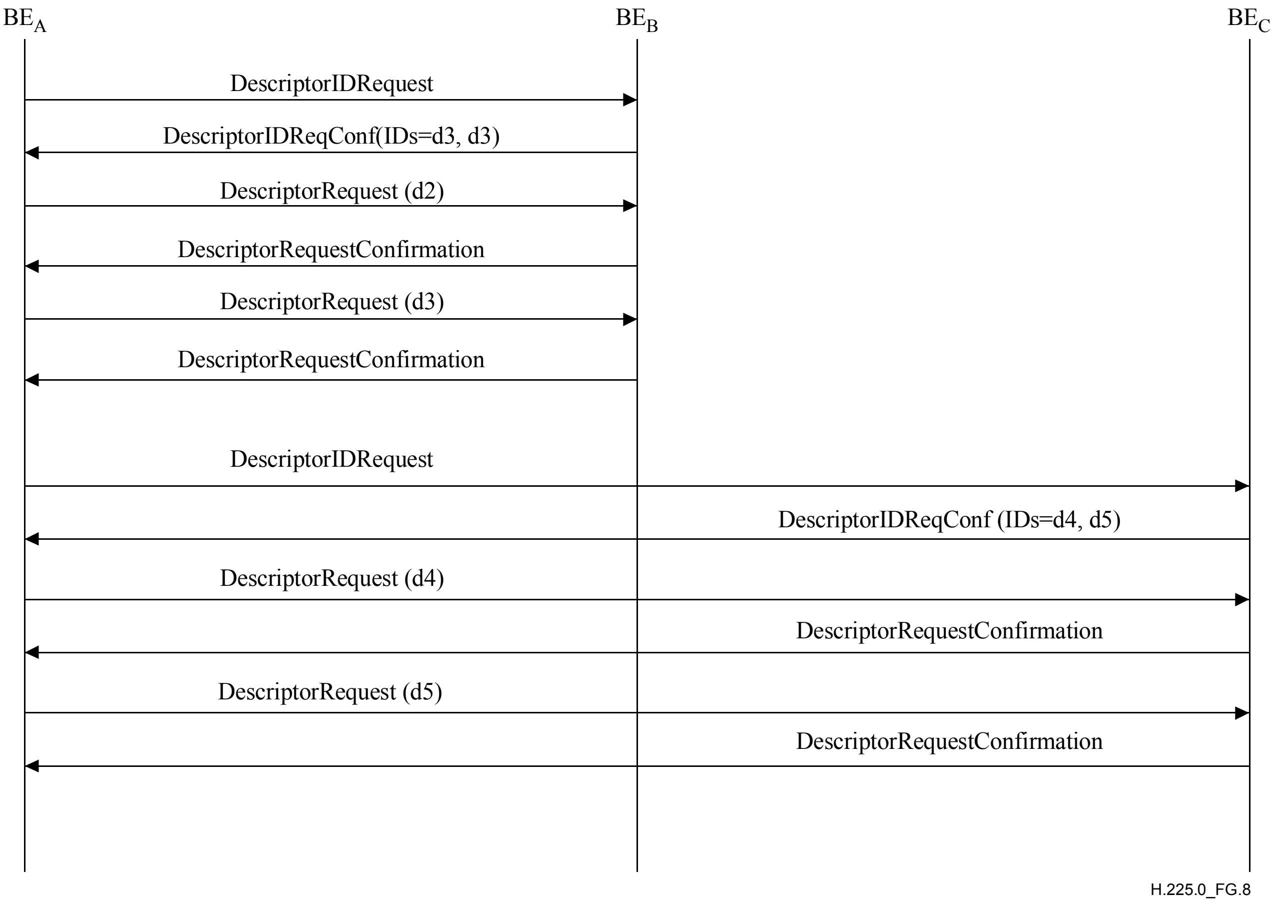

BE_A
BE_B
BE_C
DescriptorIDRequest
DescriptorIDReqConf(IDs=d3, d3)
DescriptorRequest (d2)
DescriptorRequestConfirmation
DescriptorRequest (d3)
DescriptorRequestConfirmation
DescriptorIDRequest
DescriptorIDReqConf (IDs=d4, d5)
DescriptorRequest (d4)
DescriptorRequestConfirmation
DescriptorRequest (d5)
DescriptorRequestConfirmation
H.225.0_FG.8

Figure G.8/H.225.0 – Example of descriptor exchange

Similarly, BE_B queries BE_A and BE_C, and BE_C queries BE_A and BE_B.

G.7.1.2 Placing a call

Suppose that T1 in Administrative Domain A initiates a call to 19085551515 (T2). On receipt of T1's ARQ, T1's gatekeeper sends an LRQ. A Border Element in Administrative Domain A, BE_A, has previously received zone descriptors and knows how to process the request. As shown in Figure G.9, BE_A sends an AccessRequest message to BE_B, as specified in the descriptor BE_A received from BE_B. BE_B replies back with T2's call signalling address (in this example, T2 could be any type of endpoint). T1 then sends the H.225.0 Setup message to T2's call signalling address following the normal procedures defined in ITU-T Rec. H.323 and its annexes.

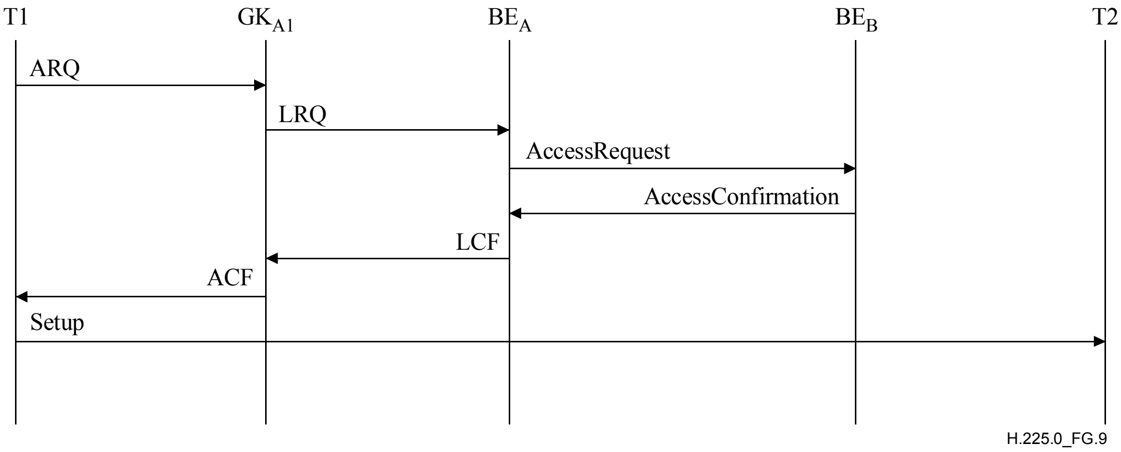


Figure G.9/H.225.0 – Example of a simple call

Now, suppose that T1 initiates a call to 19089532000. In this example, BE$_A$ has previously obtained the call signalling address of a gateway in Administrative Domain that will accept the call. As shown in Figure G.10, BE$_A$ can respond to the LRQ without any message exchange into Administrative Domain B, allowing T1 to send the Setup message directly to the gateway.

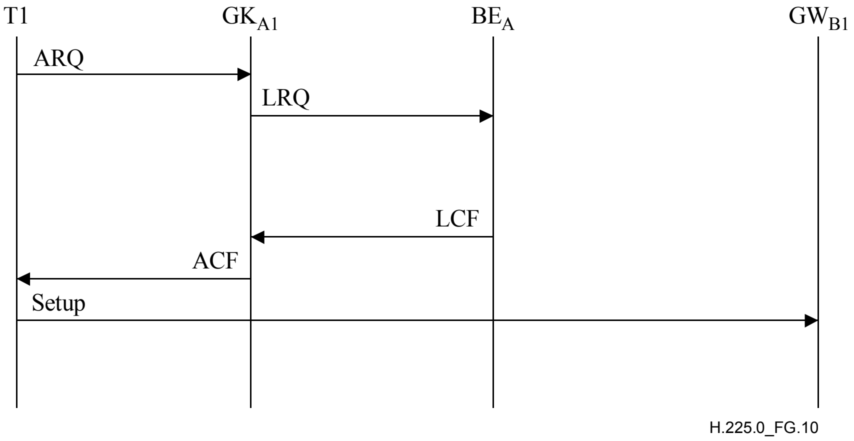


Figure G.10/H.225.0 – Example of a call with cached address

In another example, suppose that T1 initiates a call to 13035382899. Administrative Domain C has advertised its ability to accept a call to this number, and will accept call signalling through its gatekeeper in implementing the gatekeeper-routed model. As shown in Figure G.11, BE$_A$ can respond to the LRQ with an LCF that contains the call signalling address of a gatekeeper in Administrative Domain C without any message exchange into Administrative Domain C.

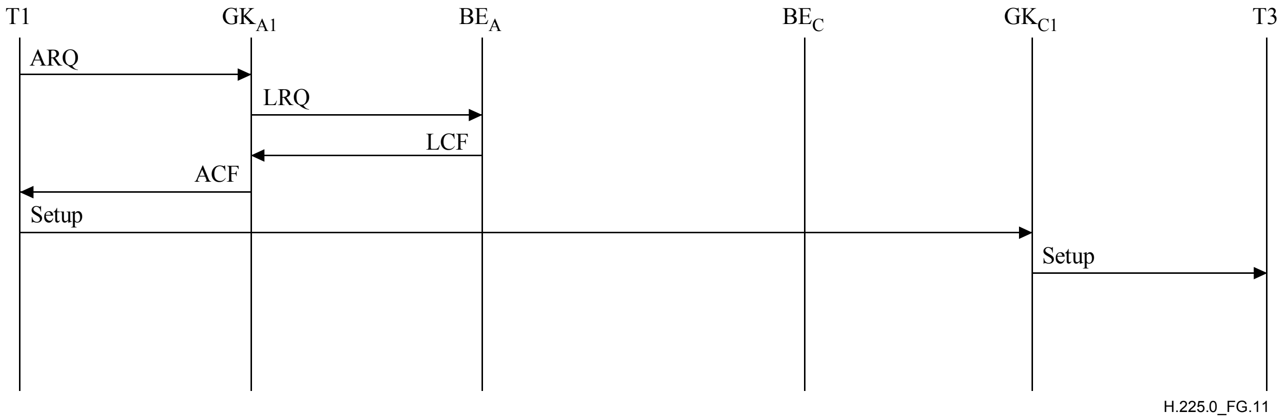


Figure G.11/H.225.0 – Example of remote gatekeeper routed call

Alternatively, T1's gatekeeper can implement the gatekeeper-routed model, as shown in Figure G.12.

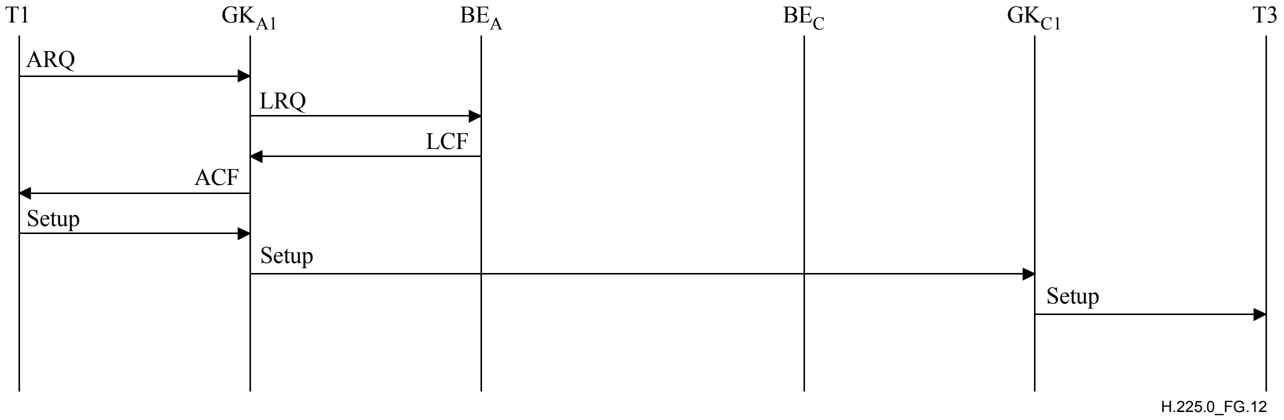


Figure G.12/H.225.0 – Example of local gatekeeper routed call

G.7.2 Clearing house

An example of a configuration using a Clearing House is shown in Figure G.13. Refer to this figure for the following examples. In this example, the Clearing House holds addressing information for all Administrative Domains for which the Clearing House provides service.

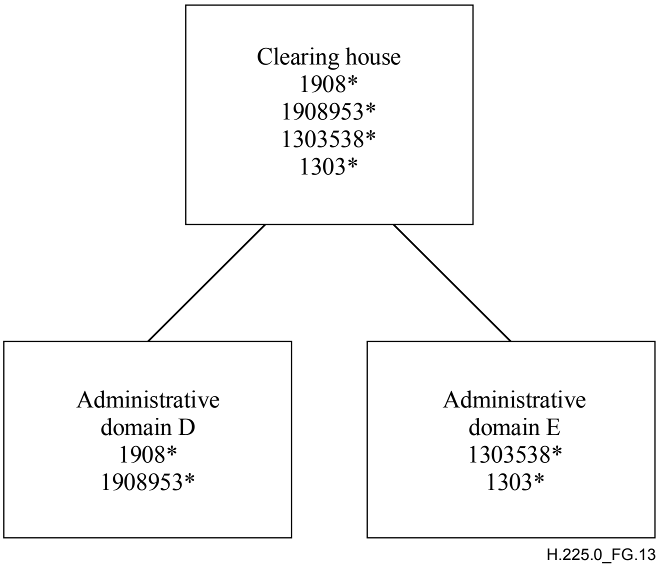


Figure G.13/H.225.0 – Sample clearing house configuration

For this example, the Border Elements in Administrative Domains D and E, and the Clearing House, contain the following information:

Administrative domain	Template definition	Comment
D	Descriptor "d1": Pattern = 1908* Transport address = BE_D Annex G address Message type = **sendAccessRequest** Descriptor "d2": Pattern = 1908953* Transport address = GW_{D1} Call Signalling address Message type = **sendSetup**	For calls to 1908*, an AccessRequest message is needed to get the destination's (i.e., a gateway) call signalling address. For calls to 1908953*, the Setup can be sent directly to this particular gateway.
E	Descriptor "d3": Pattern = 1303538* Transport address = GK_{E1} call signal address Message type = **sendSetup** Descriptor "d4": Pattern = 1303* Transport address = BE_E Annex G address Message type = **sendAccessRequest**	Calls to 1303538* will be routed through this particular gatekeeper. Calls to 1303* can be signaled directly to the destination gateway, but an AccessRequest must be sent to obtain the gateway's call signalling address.

Administrative domain	Template definition	Comment
CH	Descriptor "d1": Pattern = 1908* Transport address = BE_D Annex G address Message type = **sendAccessRequest** Descriptor "d2": Pattern = 1908953* Transport address = GW_{D1} call signalling address Message type = **sendSetup** Descriptor "d3": Pattern = 1303538* Transport address = GK_{E1} call signal address Message type = **sendSetup** Descriptor "d4": Pattern = 1303* Transport address = BE_E Annex G address Message type = **sendAccessRequest**	The Clearing House obtains descriptors from other ADs and holds this information for distribution during descriptor exchange.

G.7.2.1 Exchange of zone information

In this example, a Clearing House exchanges information with Administrative Domains that subscribe to the Clearing House's service. The Clearing House holds the information it receives from each Administrative Domain and passes this information along to other Administrative Domains. In this example, the Clearing House appears as Administrative Domain E to Administrative Domain D, while Administrative Domains D and E are not necessarily aware of each other. See Figure G.14.

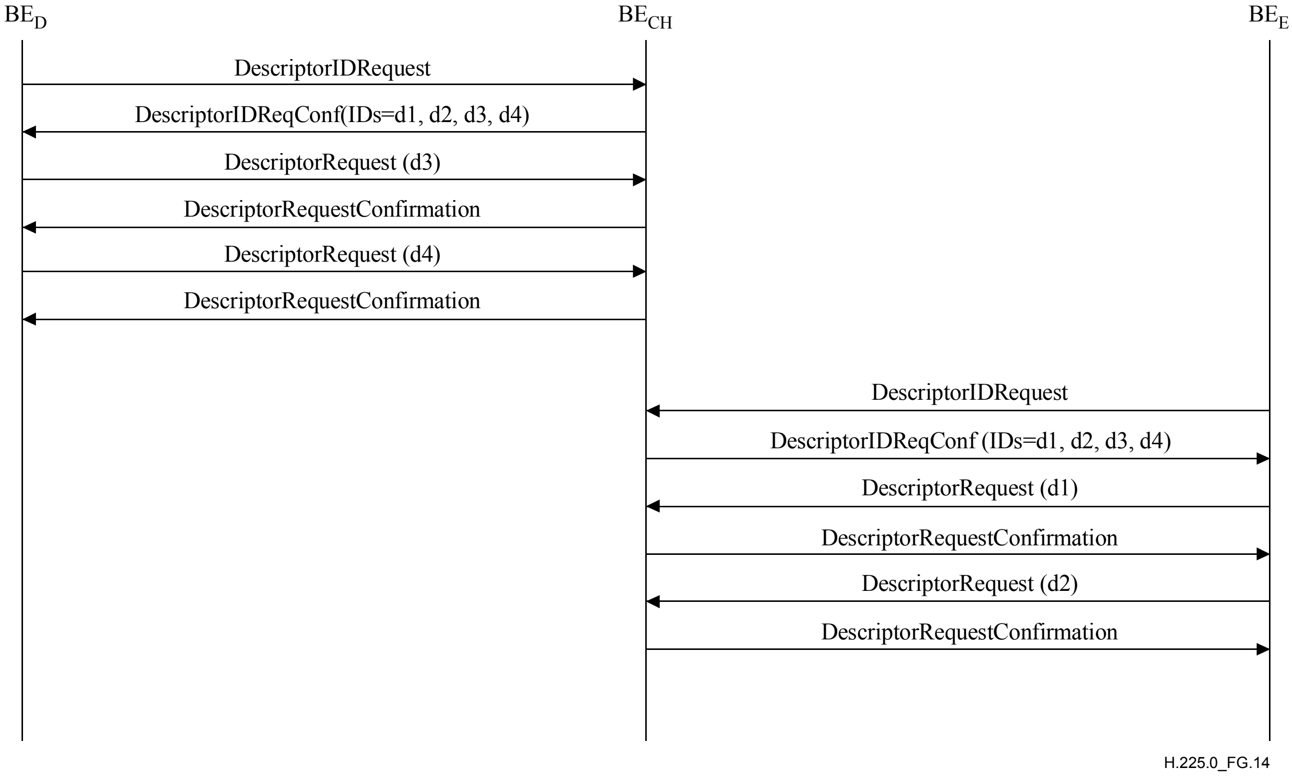

BE_D
BE_{CH}
BE_E
DescriptorIDRequest
DescriptorIDReqConf(IDs=d1, d2, d3, d4)
DescriptorRequest (d3)
DescriptorRequestConfirmation
DescriptorRequest (d4)
DescriptorRequestConfirmation
DescriptorIDRequest
DescriptorIDReqConf (IDs=d1, d2, d3, d4)
DescriptorRequest (d1)
DescriptorRequestConfirmation
DescriptorRequest (d2)
DescriptorRequestConfirmation
H.225.0_FG.14

Figure G.14/H.225.0 – Example descriptor exchange with clearing house

G.7.2.2 Placing a call

Suppose that T1 in Administrative Domain E initiates a call to 19085551515. The Border Element in Administrative Domain E has received descriptors from the Clearing House that indicate the Clearing House should be consulted for such a call. The Border Element sends an AccessRequest to the Clearing House Border Element. Based on the descriptors the Clearing House Border Element received from the Border Element in Administrative Domain D, the Clearing House Border Element sends an AccessRequest to the Border Element in Administrative Domain D. When the Clearing House Border Element returns the confirmation to the Border Element in Administrative Domain E, the confirmation contains the information sent from the Border Element in Administrative Domain D. T1's gatekeeper returns an ACF with T2's destCallSignalAddress, allowing T1 to send the Setup message to T2. See Figure G.15.

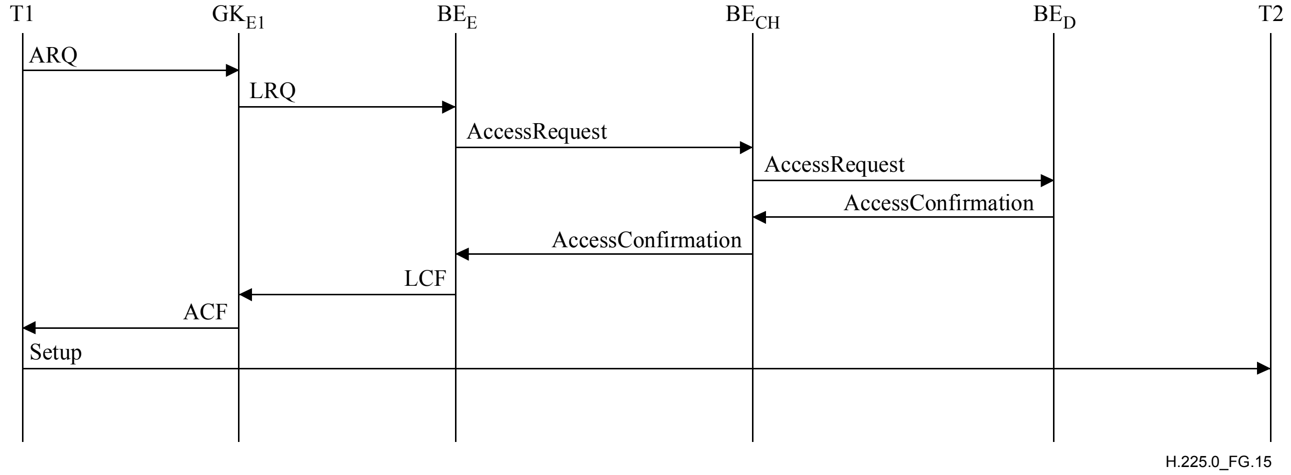


Figure G.15/H.225.0 – Example of call with clearing house

Alternatively, T1's gatekeeper could route the call signalling, as shown in Figure G.16.

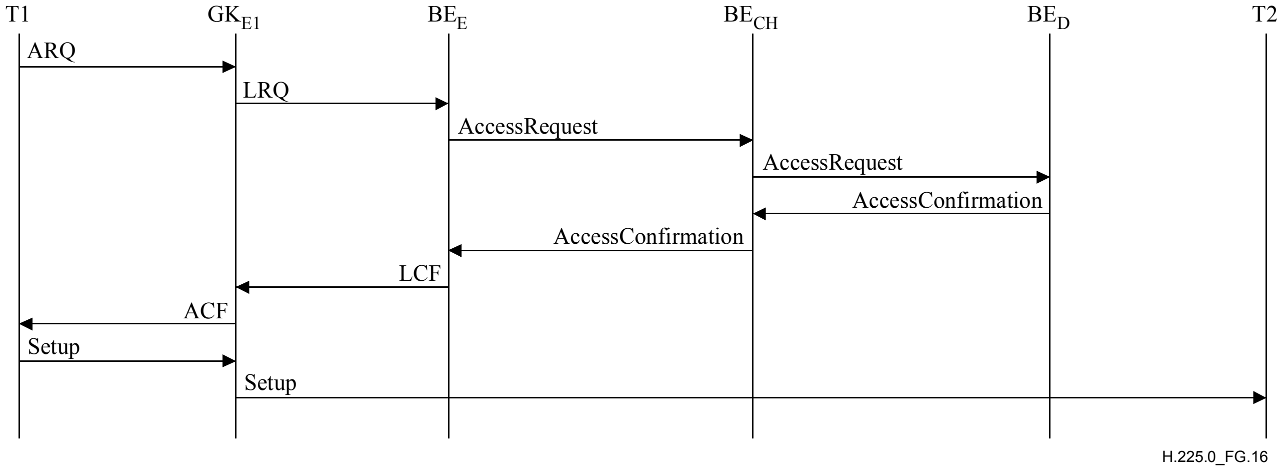


Figure G.16/H.225.0 – Example of local gatekeeper routed call with clearing house

Another possibility is for the Clearing House to respond to the Border Element in Administrative Domain E with the contact information for the Border Element in Administrative Domain D, as shown in Figure G.17.

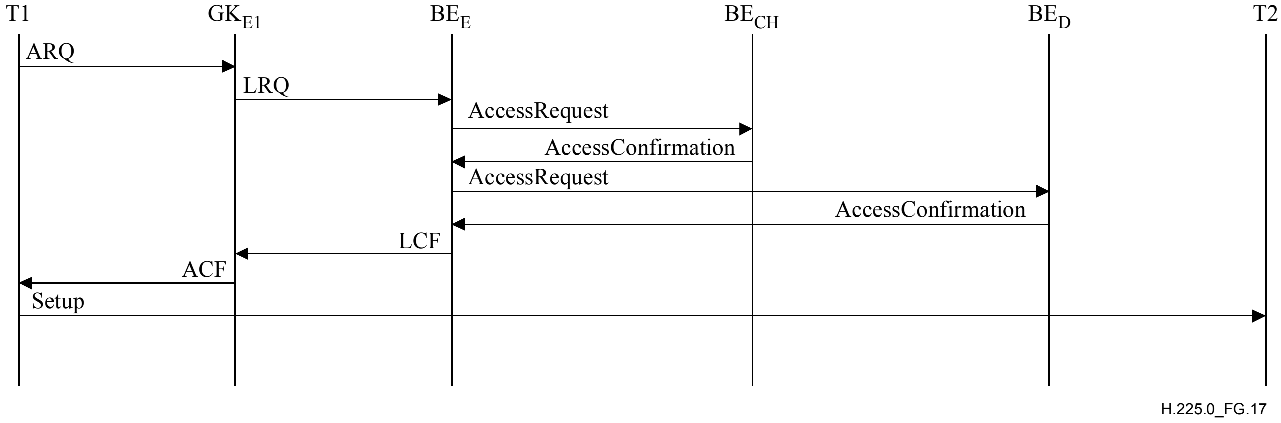


Figure G.17/H.225.0 – Example of clearing house routing using contact info for remote BE

Now suppose that T1 initiates a call to 19089532000. The descriptors previously exchanged allow the Border Element to return the call signalling address to T1 without consulting the Clearing House, as shown in Figure G.18.

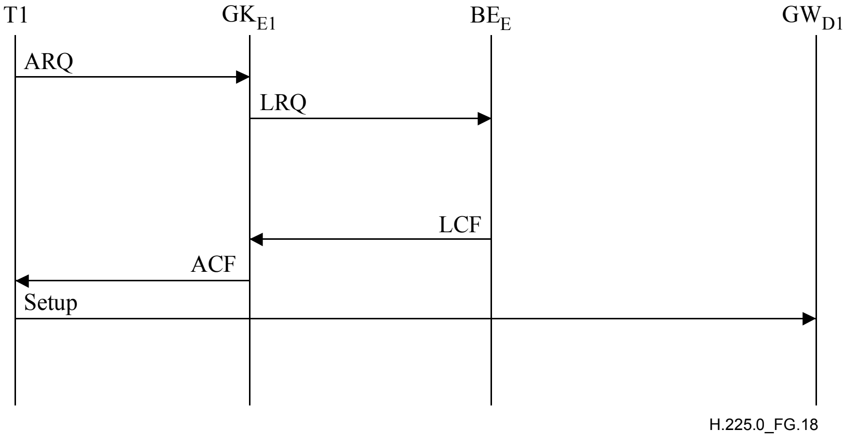


Figure G.18/H.225.0 – Example of call using cached descriptor in local BE

Next, consider a scenario where T1 initiates a call to 13035382899. The Border Element in Administrative Domain E had previously advertised that calls to 1303538* could be routed directly to a gatekeeper in Administrative Domain E without need for an AccessRequest message, as shown in Figure G.19. (This advertisement does not indicate that the entity is a gatekeeper, only that a Setup message could be sent to a specified address.) The Border Element in Administrative Domain D received this information from the Clearing House, assuming the Clearing House in this example does not have a requirement to provide address resolution for these calls.

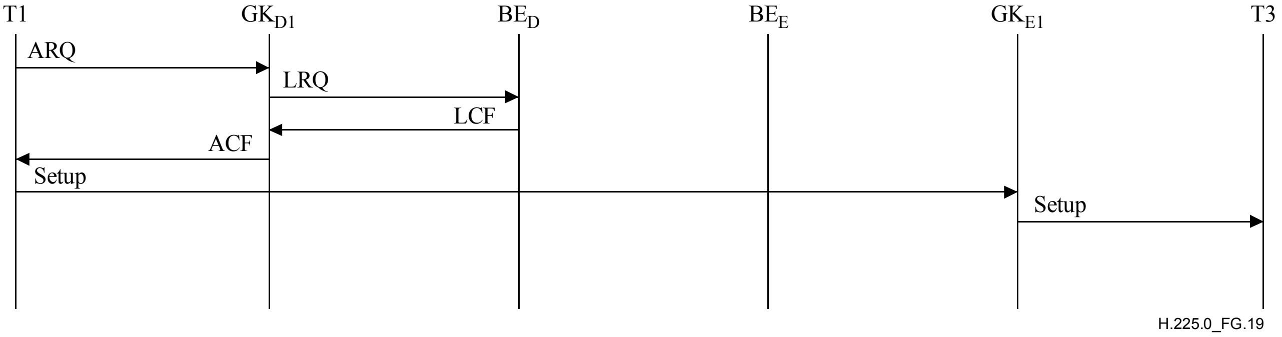


Figure G.19/H.225.0 – Example of GK routed call using cached descriptor

Recall that a Border Element may be combined with a gatekeeper, and may also route calls in the gatekeeper-routed model. An alternative signalling example is shown in Figure G.20. It is also possible to use the Border Element as a routing gatekeeper into an Administrative Domain if the descriptors are so configured.

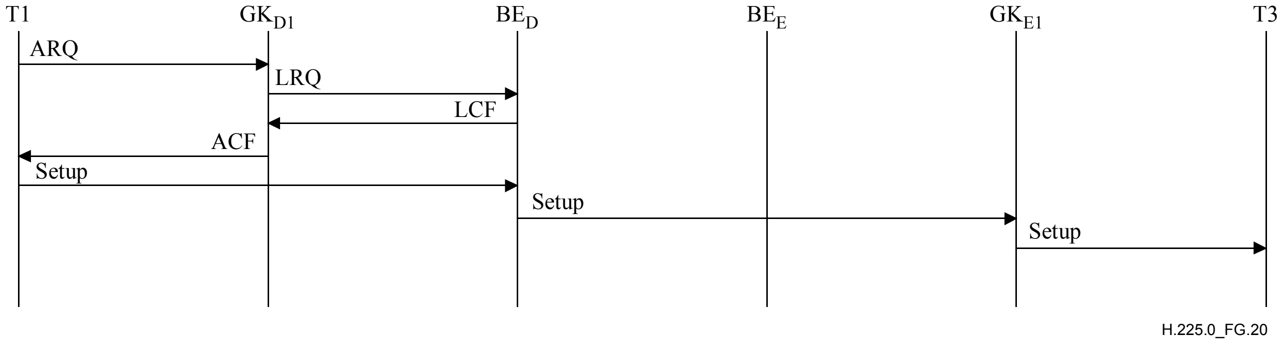


Figure G.20/H.225.0 – Example of call with combined BE/routing GK

In the example of Figure G.21, the Clearing House validates the call for the terminating Administrative Domain. The Clearing House also requires both originating and terminating Border Elements to send UsageIndications for the call.

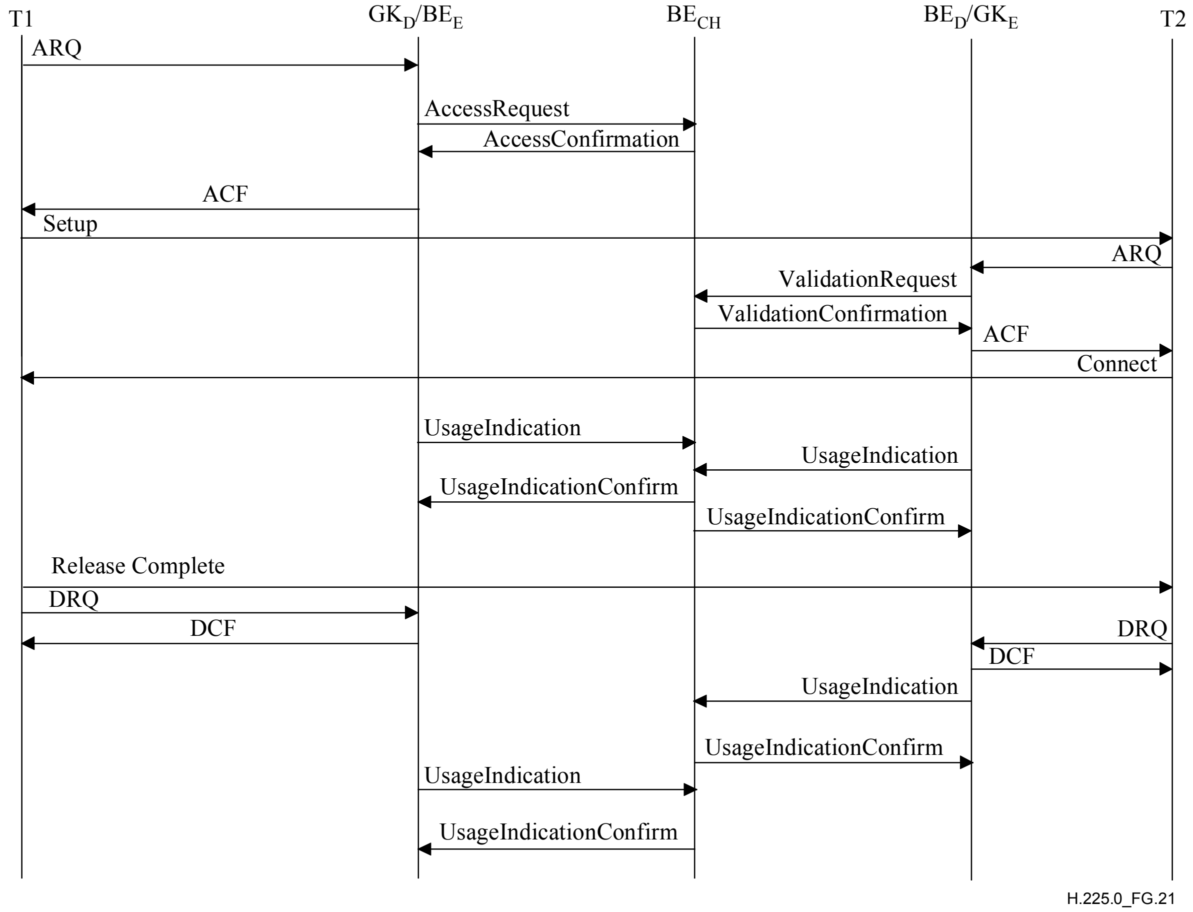


Figure G.21/H.225.0 – Call validation and usage reporting example with clearing house

G.8 Annex G profiles

G.8.1 Introduction

ITU-T Rec. H.501 offers a rich set of messages and fields that Annex G/H.225.0 may use for interaction between Administrative Domains and between Peer Elements within a single Administrative Domain. Many of the messages and fields are optional and can be used in a variety of ways to implement different services or service options. This clause specifies implementation profiles that define the messages, fields and procedures that are required in order to claim compliance with a specific profile.

G.8.1.1 Profile signalling and negotiation

The H.323 generic extensible framework may be used by a Peer Element to signal to another Peer Element the set of profiles that it needs for a transaction to be successful, the set of profiles it desires to use, and the set of profiles it supports. This profile negotiation signalling may be done either in an individual message exchange (e.g., in an AccessRequest/AccessConfirmation exchange), or during the establishment of a service relationship. Note that the establishment of a service relationship between two Peer Elements may not be required by a profile.

G.8.1.1.1 Processing by the requesting entity

A requesting entity (a Peer Element) uses the elements in the **`FeatureSet`** structure to specify the various profiles it requires. It specifies the set of profiles that it needs using the **`neededFeatures`** field, the set of profiles that it desires using the **`desiredFeatures`** field, and the set of profiles that it supports in the **`supportedFeatures`** field. All three of these fields are in the **`FeatureSet`** structure.

In response to its request, a requesting entity should receive either a confirmation or a rejection message.

If the request is rejected, the responding entity may have included a set of **`neededFeatures`** that the requesting entity must support in order for the request to be successful. If this is the case and the requesting entity supports the needed features (e.g., a specific profile), the requesting entity may re-issue a request specifying support for the profile needed by the responding entity.

If the request is accepted, special procedures need to be applied to ensure that the negotiation operates in a backwards-compatible manner. This is done by the requesting entity checking that the profile that it specified as needed are listed as **`supportedFeatures`** in the response. If a requesting entity does not observe the profiles it needs in the response message's **`supportedFeatures`** field, then it shall assume that the responding entity does not support the profiles that it needs. If the requesting entity determines that it cannot continue under these circumstances, then it shall undo the operation it was trying to perform (i.e., send a ServiceRelease message if it originally sent a ServiceRequest message), so that the state in the responding entity is rolled back.

G.8.1.1.2 Processing by the responding entity

The responding entity looks at the profiles specified in the **`neededFeatures`** field of the request to determine if it can accept the request. It also looks in the **`neededFeatures`**, **`desiredFeatures`** and **`supportedFeatures`** fields to determine whether the profiles needed by it are supported by the requesting entity.

If the responding entity determines that the necessary sets of profiles are supported by both entities, then the responding entity may acknowledge the request. The responding entity lists the set of profiles that it chooses to support in the **`supportedFeatures`** field of it reply. If the request is accepted, then all of the **`neededFeatures`** from the request must be included in the **`supportedFeatures`** field of the reply. The responding entity may also include **`desiredFeatures`**.

If the responding entity needs additional profiles to be supported by the requesting entity, it shall reject the request. If it wishes to declare which profiles must be supported for the request to be successful, this should be specified using the **`neededFeatures`** field of the reject message. The responding entity may also include any **`desiredFeatures`** and **`supportedFeatures`** in the reject message.

G.8.1.1.3 Identifiers

The following identifier is used within a FeatureDescriptor to specify that the FeatureDescriptor applies to Annex G/H.225.0 profiles.

Value	Description
idAnnexGProfiles	This identifier is used in the "id" field of a FeatureDescriptor to indicate that the FeatureDescriptor is describing the Annex G profiles needed/desired/supported.

The following table contains a list of the identifiers used within the generic extensibility framework that are relevant to Annex G/H.225.0.

Standard INTEGER Value	Description
0	Identifier within a FeatureDescriptor indicating that the FeatureDescriptor is describing Annex G/H.225.0 profiles
1	Identifier within an EnumeratedParameter that identifies Annex G/H.225.0 Profile "A"

G.8.2 Profile "A": Interzone call routing to a trusted gatekeeper

This profile specifies a simple intra-domain service: per-call queries to another trusted zone for endpoint determination where the Annex G signalling address of the trusted zones is statically provisioned. This is one of the simplest uses of Annex G and is similar to the use of RAS LRQ to query another zone for an endpoint. The same profile may be used to query a trusted Peer Element, which returns routes from domain-wide knowledge or obtains them by further Annex G queries.

G.8.2.1 Required messages

Entities complying with this profile shall support the messages indicated as "Mandatory" in the following table:

Message	Transmit (**M**andatory, **O**ptional, **R**ecommended)	Receive and act on (**M**andatory, **O**ptional, **R**ecommended)
ServiceRequest	O	M (Note 1)
ServiceConfirmation	O	O
ServiceRejection	M	O
ServiceRelease	O	O
DescriptorRequest	O	M (Note 1)
DescriptorConfirmation	R (Note 2)	O
DescriptorRejection	M	O
DescriptorIdRequest	O	M (Note 1)
DescriptorIdConfirmation	R (Note 3)	O
DescriptorIdRejection	M	O
DescriptorUpdate	O	M (Note 4)
DescriptorUpdateAck	M	O
AccessRequest	M	M
AccessConfirmation	M	M
AccessRejection	M	M
RequestInProgress	M	M
NonStandardRequest	O	M

Message	**Transmit** (**M**andatory, **O**ptional, **R**ecommended)	**Receive and act on** (**M**andatory, **O**ptional, **R**ecommended)
NonStandardConfirmation	O	O
NonStandardRejection	M	O
UnknownMessageResponse	M	M
UsageRequest	O	M (Note 1)
UsageConfirmation	O	O
UsageRejection	M	O
UsageIndication	O	M (Note 1)
UsageIndicationConfirmation	O	O
UsageIndicationRejection	M	O
ValidationRequest	O	M (Note 1)
ValidationConfirmation	O	O
ValidationRejection	M	O
NOTE 1 – Shall be received and as a minimum rejected. NOTE 2 – It is recommended that an entity return as a minimum a single descriptor for a template with SendAccessRequest pointing to itself. NOTE 3 – It is recommended that an entity return as a minimum a single descriptor for a template with SendAccessRequest pointing to itself. NOTE 4 – Shall be received and Acknowledged, but need not be processed.		

G.8.2.2 Required fields

All fields defined as mandatory by ITU-T Rec. H.501 are also mandatory within this profile.

Entities complying with this profile shall also support the fields specified in the following table.

Other fields defined as optional by ITU-T Rec. H.501 may optionally be present.

Message or structure	Required field	Comment
AccessRequest message	`destinationInfo`	One address containing the fully-qualified E.164 address of the destination
	`sourceInfo`	Includes the domainInfo and endpointType
	`callInfo`	
AccessConfirmation message	`templates`	If any templates are present, then there is one template per each termination gateway/gatekeeper
	`partialResponse`	Set to FALSE
AddressTemplate structure	`pattern`	One specific pattern is present containing the E.164 number
	`routeInfo`	One instance present
	`timeToLive`	

Message or structure	Required field	Comment
RouteInformation structure	`messageType`	Present
	`callSpecific`	Set to FALSE
	`contacts`	One instance present
	`type`	Must be present if messageType = **sendSetup**
ContactInformation structure	`transportAddress`	The IP address of the gateway/gatekeeper
	`priority`	

G.8.2.3 Required procedures

In this profile, entities may use the static discovery procedures of Annex G (see G.6.3.1) and so will have a configured list of Peer Elements or gatekeepers to which requests can be sent. This list may contain alternates to be used only when the primary element cannot be reached or may simply add the alternates (if any) to the list.

Entities may also use the dynamic discovery procedures of Annex G (see G.6.3.2).

Entities shall send an AccessRequest message to a selected Peer Element or gatekeeper for each call. If more than one Peer Element or gatekeeper is available to be queried for a given call, it is not specified whether they must be queried in sequence or may be queried in parallel. This choice is left to the requesting entity.

The reply will have zero or more templates. `timeToLive` may be set to 60 seconds or less to indicate that it may not be used for another call.

To improve interoperation with more general peers, it is suggested that in the case that the Peer Element does not implement descriptor support, it should follow the following procedures:

- If a DescriptorIDRequest message is received, the Peer Element should return a DescriptorIDConfirmation message containing a single `DescriptorInfo`. This `DescriptorInfo` describes a descriptor containing a single template specifying `sendAccessRequest` pointing to the Peer Element itself.
- If a DescriptorRequest message is received, the Peer Element should return a DescriptorConfirmation message containing a single descriptor. This descriptor shall contain a single template specifying `sendAccessRequest` pointing to the Peer Element itself.

G.8.2.4 Identifiers for Profile "A"

The following identifier is used within an `EnumeratedParameter` to specify that the `EnumeratedParameter` specifies Annex G/H.225 Profile A.

Value	Description
idAnnexGProfileA	This identifier is used in the "id" field of an `EnumeratedParameter` to indicate that Annex G Profile A is needed/desired/supported. Note that the "content" field of the `EnumeratedParameter` is not present.

Annex H

H.225.0 message syntax (ASN.1)

This Recommendation defines protocols for RAS (essentially a gatekeeper protocol) and call signalling (essentially protocol data units which reside in a User-user information element). These protocols are defined together in the following ASN.1 tree. Semantic definitions for the messages and various elements appear in previous clauses.

```
H323-MESSAGES DEFINITIONS AUTOMATIC TAGS ::=
BEGIN


IMPORTS
          SIGNED{},
          ENCRYPTED{},
          HASHED{},
          ChallengeString,
          TimeStamp,
          RandomVal,
          Password,
          EncodedPwdCertToken,
          ClearToken,
          CryptoToken,
          AuthenticationMechanism
     FROM H235-SECURITY-MESSAGES
          DataProtocolCapability,
          T38FaxProfile
     FROM MULTIMEDIA-SYSTEM-CONTROL;
H323-UserInformation ::= SEQUENCE     -- root for all H.225.0 call signalling
                                      -- messages
{
     h323-uu-pdu   H323-UU-PDU,
     user-data     SEQUENCE
     {
          protocol-discriminator      INTEGER   (0..255),
          user-information            OCTET STRING (SIZE(1..131)),
          ...
     } OPTIONAL,
     ...
}

H323-UU-PDU ::= SEQUENCE
{
     h323-message-body   CHOICE
     {
          setup               Setup-UUIE,
          callProceeding      CallProceeding-UUIE,
          connect             Connect-UUIE,
          alerting            Alerting-UUIE,
          information         Information-UUIE,
          releaseComplete     ReleaseComplete-UUIE,
          facility            Facility-UUIE,
          ...,
          progress       Progress-UUIE,
          empty          NULL,     -- used when a Facility message is sent,
                                   -- but the Facility-UUIE is not to be invoked
                                   -- (possible when transporting supplementary
                                   -- services messages in versions prior to
                                   -- H.225.0 version 4)
```

```
        status                  Status-UUIE,
        statusInquiry           StatusInquiry-UUIE,
        setupAcknowledge        SetupAcknowledge-UUIE,
        notify                  Notify-UUIE
    },
    nonStandardData             NonStandardParameter OPTIONAL,
    ...,
    h4501SupplementaryService        SEQUENCE OF OCTET STRING OPTIONAL,
                                -- each sequence of octet string is defined as one
                                -- H4501SupplementaryService APDU as defined in
                                -- Table 3/H.450.1

    h245Tunnelling              BOOLEAN,
                                -- if TRUE, tunnelling of H.245 messages is enabled
    h245Control                 SEQUENCE OF OCTET STRING OPTIONAL,
    nonStandardControl          SEQUENCE OF NonStandardParameter OPTIONAL,
    callLinkage                 CallLinkage OPTIONAL,
    tunnelledSignallingMessage       SEQUENCE
    {
        tunnelledProtocolID          TunnelledProtocol, -- tunnelled signalling
    protocol ID
        messageContent               SEQUENCE OF OCTET STRING, -- sequence of entire
                                                           -- message(s)
        tunnellingRequired           NULL OPTIONAL,
        nonStandardData              NonStandardParameter OPTIONAL,
        ...
    } OPTIONAL,
    provisionalRespToH245Tunnelling       NULL OPTIONAL,
    stimulusControl                       StimulusControl OPTIONAL,
    genericData                           SEQUENCE OF GenericData OPTIONAL
}

StimulusControl ::= SEQUENCE
{
    nonStandard                           NonStandardParameter OPTIONAL,
    isText                                NULL OPTIONAL,
    h248Message                           OCTET STRING OPTIONAL,
    ...
}

Alerting-UUIE ::= SEQUENCE
{
    protocolIdentifier               ProtocolIdentifier,
    destinationInfo                  EndpointType,
    h245Address                      TransportAddress OPTIONAL,
    ...,
    callIdentifier                   CallIdentifier,
    h245SecurityMode                 H245Security OPTIONAL,
    tokens                           SEQUENCE OF ClearToken OPTIONAL,
    cryptoTokens                     SEQUENCE OF CryptoH323Token OPTIONAL,
    fastStart                        SEQUENCE OF OCTET STRING OPTIONAL,
    multipleCalls                    BOOLEAN,
    maintainConnection               BOOLEAN,
    alertingAddress                  SEQUENCE OF AliasAddress OPTIONAL,
    presentationIndicator            PresentationIndicator OPTIONAL,
    screeningIndicator               ScreeningIndicator OPTIONAL,
    fastConnectRefused               NULL OPTIONAL,
    serviceControl                   SEQUENCE OF ServiceControlSession OPTIONAL,
    capacity                         CallCapacity OPTIONAL,
    featureSet                       FeatureSet OPTIONAL
}
```

```
CallProceeding-UUIE ::= SEQUENCE
{
    protocolIdentifier          ProtocolIdentifier,
    destinationInfo             EndpointType,
    h245Address                 TransportAddress OPTIONAL,
    ...,
    callIdentifier              CallIdentifier,
    h245SecurityMode            H245Security OPTIONAL,
    tokens                      SEQUENCE OF ClearToken OPTIONAL,
    cryptoTokens                SEQUENCE OF CryptoH323Token OPTIONAL,
    fastStart                   SEQUENCE OF OCTET STRING OPTIONAL,
    multipleCalls               BOOLEAN,
    maintainConnection          BOOLEAN,
    fastConnectRefused          NULL OPTIONAL,
    featureSet                  FeatureSet OPTIONAL
}

Connect-UUIE ::= SEQUENCE
{
    protocolIdentifier          ProtocolIdentifier,
    h245Address                 TransportAddress OPTIONAL,
    destinationInfo             EndpointType,
    conferenceID                ConferenceIdentifier,
    ...,
    callIdentifier              CallIdentifier,
    h245SecurityMode            H245Security OPTIONAL,
    tokens                      SEQUENCE OF ClearToken OPTIONAL,
    cryptoTokens                SEQUENCE OF CryptoH323Token OPTIONAL,
    fastStart                   SEQUENCE OF OCTET STRING OPTIONAL,
    multipleCalls               BOOLEAN,
    maintainConnection          BOOLEAN,
    language                    SEQUENCE OF IA5String (SIZE (1..32)) OPTIONAL,
                                -- RFC 1766 language tag
    connectedAddress            SEQUENCE OF AliasAddress OPTIONAL,
    presentationIndicator       PresentationIndicator OPTIONAL,
    screeningIndicator          ScreeningIndicator OPTIONAL,
    fastConnectRefused          NULL OPTIONAL,
    serviceControl              SEQUENCE OF ServiceControlSession OPTIONAL,
    capacity                    CallCapacity OPTIONAL,
    featureSet                  FeatureSet OPTIONAL
}

Information-UUIE ::=SEQUENCE
{
    protocolIdentifier          ProtocolIdentifier,
    ...,
    callIdentifier              CallIdentifier,
    tokens                      SEQUENCE OF ClearToken OPTIONAL,
    cryptoTokens                SEQUENCE OF CryptoH323Token OPTIONAL,
    fastStart                   SEQUENCE OF OCTET STRING OPTIONAL,
    fastConnectRefused          NULL OPTIONAL,
    circuitInfo                 CircuitInfo OPTIONAL
}

ReleaseComplete-UUIE ::= SEQUENCE
{
    protocolIdentifier          ProtocolIdentifier,
    reason                      ReleaseCompleteReason OPTIONAL,
    ...,
    callIdentifier              CallIdentifier,
    tokens                      SEQUENCE OF ClearToken OPTIONAL,
    cryptoTokens                SEQUENCE OF CryptoH323Token OPTIONAL,
    busyAddress                 SEQUENCE OF AliasAddress OPTIONAL,
    presentationIndicator       PresentationIndicator OPTIONAL,
    screeningIndicator          ScreeningIndicator OPTIONAL,
    capacity                    CallCapacity OPTIONAL,
    serviceControl              SEQUENCE OF ServiceControlSession OPTIONAL,
    featureSet                  FeatureSet OPTIONAL
}

ReleaseCompleteReason ::= CHOICE
{
    noBandwidth                 NULL, -- bandwidth taken away or ARQ denied
    gatekeeperResources         NULL, -- exhausted
    unreachableDestination      NULL, -- no transport path to the destination
    destinationRejection        NULL, -- rejected at destination
    invalidRevision             NULL,
    noPermission                NULL, -- called party's gatekeeper rejects
    unreachableGatekeeper       NULL, -- terminal cannot reach gatekeeper
                                      -- for ARQ
    gatewayResources            NULL,
    badFormatAddress            NULL,
    adaptiveBusy                NULL, -- call is dropping due to LAN crowding
    inConf                      NULL, -- called party busy
    undefinedReason             NULL,
    ...,
    facilityCallDeflection      NULL, -- call was deflected using a Facility
                                      -- message
    securityDenied              NULL, -- incompatible security settings
    calledPartyNotRegistered    NULL, -- used by gatekeeper when endpoint has
                                      -- preGrantedARQ to bypass ARQ/ACF
    callerNotRegistered         NULL, -- used by gatekeeper when endpoint has
                                      -- preGrantedARQ to bypass ARQ/ACF
    newConnectionNeeded         NULL, -- indicates that the Setup was not
                                      -- accepted on this connection, but that
                                      -- the Setup may be accepted on
                                      -- a new connection
    nonStandardReason           NonStandardParameter,
    replaceWithConferenceInvite ConferenceIdentifier, -- call dropped due to
                                                      -- subsequent invitation
                                                      -- to a conference
                                                      -- (see 8.4.3.8/H.323)
    genericDataReason           NULL,
    neededFeatureNotSupported   NULL,
    tunnelledSignallingRejected NULL,
    invalidCID                  NULL,
    securityError               SecurityErrors,
    hopCountExceeded            NULL
}

Setup-UUIE ::= SEQUENCE
{
    protocolIdentifier      ProtocolIdentifier,
    h245Address             TransportAddress OPTIONAL,
    sourceAddress           SEQUENCE OF AliasAddress OPTIONAL,
    sourceInfo              EndpointType,
    destinationAddress      SEQUENCE OF AliasAddress OPTIONAL,
    destCallSignalAddress   TransportAddress OPTIONAL,
    destExtraCallInfo       SEQUENCE OF AliasAddress OPTIONAL,
    destExtraCRV            SEQUENCE OF CallReferenceValue OPTIONAL,
    activeMC                BOOLEAN,
    conferenceID            ConferenceIdentifier,
    conferenceGoal          CHOICE
```

```
     {
          create             NULL,
          join               NULL,
          invite             NULL,
          ...,
          capability-negotiation                NULL,
          callIndependentSupplementaryService   NULL
     },
     callServices                 QseriesOptions OPTIONAL,
     callType                     CallType,
     ...,
     sourceCallSignalAddress      TransportAddress OPTIONAL,
     remoteExtensionAddress       AliasAddress OPTIONAL,
     callIdentifier               CallIdentifier,
     h245SecurityCapability       SEQUENCE OF H245Security OPTIONAL,
     tokens                       SEQUENCE OF ClearToken OPTIONAL,
     cryptoTokens                 SEQUENCE OF CryptoH323Token OPTIONAL,
     fastStart                    SEQUENCE OF OCTET STRING OPTIONAL,
     mediaWaitForConnect          BOOLEAN,
     canOverlapSend               BOOLEAN,
     endpointIdentifier           EndpointIdentifier OPTIONAL,
     multipleCalls                BOOLEAN,
     maintainConnection           BOOLEAN,
     connectionParameters         SEQUENCE  -- additional gateway parameters
     {
          connectionType               ScnConnectionType,
          numberOfScnConnections       INTEGER (0..65535),
          connectionAggregation        ScnConnectionAggregation,
          ...
     } OPTIONAL,
     language                     SEQUENCE OF IA5String (SIZE (1..32)) OPTIONAL,
                                  -- RFC1766 language tag
     presentationIndicator        PresentationIndicator OPTIONAL,
     screeningIndicator           ScreeningIndicator OPTIONAL,
     serviceControl               SEQUENCE OF ServiceControlSession OPTIONAL,
     symmetricOperationRequired   NULL OPTIONAL,
     capacity                     CallCapacity OPTIONAL,
     circuitInfo                  CircuitInfo OPTIONAL,
     desiredProtocols             SEQUENCE OF SupportedProtocols OPTIONAL,
     neededFeatures               SEQUENCE OF FeatureDescriptor OPTIONAL,
     desiredFeatures              SEQUENCE OF FeatureDescriptor OPTIONAL,
     supportedFeatures            SEQUENCE OF FeatureDescriptor OPTIONAL,
     parallelH245Control          SEQUENCE OF OCTET STRING OPTIONAL,
     additionalSourceAddresses    SEQUENCE OF ExtendedAliasAddress OPTIONAL,
     hopCount                     INTEGER (1..31) OPTIONAL
}

ScnConnectionType ::= CHOICE
{
     unknown     NULL,  -- should be seleceted when connection type is unknown
     bChannel    NULL,  -- each individual connection on the SCN is 64 kbit/s.
                          -- Note that where SCN delivers 56 kbit/s usable data,
                          -- the actual bandwidth allocated on SCN is still
                          -- 64 kbit/s.
     hybrid2x64  NULL,  -- each connection is a 128 kbit/s hybrid call
     hybrid384   NULL,  -- each connection is an H0 (384 kbit/s) hybrid call
     hybrid1536  NULL,  -- each connection is an H11 (1536 kbit/s) hybrid call
     hybrid1920  NULL,  -- each connection is an H12 (1920 kbit/s) hybrid call
```

```
    multirate   NULL,  -- bandwidth supplied by SCN using multirate.
                         -- In this case, the information transfer rate octet in
                         -- the bearer capability shall be set to multirate and
                         -- the rate multiplier octet shall denote the number
                         -- of B channels.
    ...
}

ScnConnectionAggregation ::= CHOICE
{
    auto           NULL,     -- aggregation mechanism is unknown
    none           NULL,     -- call produced using a single SCN connection
    h221           NULL,     -- use H.221 framing to aggregate the connections
    bonded-mode1   NULL,     -- use ISO/IEC 13871 bonding mode 1.
                             -- Use bonded-mode1 to signal a bonded call if the
                             -- precise bonding mode to be used is unknown.
    bonded-mode2   NULL,     -- use ISO/IEC 13871 bonding mode 2
    bonded-mode3   NULL,     -- use ISO/IEC 13871 bonding mode 3
    ...
}

PresentationIndicator ::= CHOICE
{
    presentationAllowed          NULL,
    presentationRestricted       NULL,
    addressNotAvailable          NULL,
    ...
}

ScreeningIndicator ::= ENUMERATED
{
    userProvidedNotScreened (0),
         -- number was provided by a remote user
         -- and has not been screened by a gatekeeper
    userProvidedVerifiedAndPassed (1),
         -- number was provided by user
         -- equipment (or by a remote network), and has
         -- been screened by a gatekeeper
    userProvidedVerifiedAndFailed (2),
         -- number was provided by user
         -- equipment (or by a remote network), and the
         -- gatekeeper has determined that the
         -- information is incorrect
    networkProvided (3),
         -- number was provided by a gatekeeper
    ...
}

Facility-UUIE ::= SEQUENCE
{
    protocolIdentifier           ProtocolIdentifier,
    alternativeAddress           TransportAddress OPTIONAL,
    alternativeAliasAddress      SEQUENCE OF AliasAddress OPTIONAL,
    conferenceID                 ConferenceIdentifier OPTIONAL,
    reason                       FacilityReason,
    ...,
    callIdentifier               CallIdentifier,
    destExtraCallInfo            SEQUENCE OF AliasAddress OPTIONAL,
    remoteExtensionAddress       AliasAddress OPTIONAL,
    tokens                       SEQUENCE OF ClearToken OPTIONAL,
    cryptoTokens                 SEQUENCE OF CryptoH323Token OPTIONAL,
    conferences                  SEQUENCE OF ConferenceList OPTIONAL,
    h245Address                  TransportAddress OPTIONAL,
    fastStart                    SEQUENCE OF OCTET STRING OPTIONAL,
    multipleCalls                BOOLEAN,
    maintainConnection           BOOLEAN,
    fastConnectRefused           NULL OPTIONAL,
    serviceControl               SEQUENCE OF ServiceControlSession OPTIONAL,
    circuitInfo                  CircuitInfo OPTIONAL,
    featureSet                   FeatureSet OPTIONAL,
    destinationInfo              EndpointType OPTIONAL,
    h245SecurityMode             H245Security OPTIONAL
}

ConferenceList ::= SEQUENCE
{
    conferenceID                 ConferenceIdentifier OPTIONAL,
    conferenceAlias              AliasAddress OPTIONAL,
    nonStandardData              NonStandardParameter OPTIONAL,
    ...
}

FacilityReason ::= CHOICE
{
    routeCallToGatekeeper   NULL,     -- call must use gatekeeper model
                                      -- gatekeeper is alternativeAddress
    callForwarded           NULL,
    routeCallToMC           NULL,
    undefinedReason         NULL,
    ...,
    conferenceListChoice    NULL,
    startH245               NULL,     -- recipient should connect to h245Address
    noH245                  NULL,     -- endpoint does not support H.245
    newTokens               NULL,
    featureSetUpdate        NULL,
    forwardedElements       NULL,
    transportedInformation  NULL
}

Progress-UUIE ::= SEQUENCE
{
    protocolIdentifier      ProtocolIdentifier,
    destinationInfo         EndpointType,
    h245Address             TransportAddress OPTIONAL,
    callIdentifier          CallIdentifier,
    h245SecurityMode        H245Security OPTIONAL,
    tokens                  SEQUENCE OF ClearToken OPTIONAL,
    cryptoTokens            SEQUENCE OF CryptoH323Token OPTIONAL,
    fastStart               SEQUENCE OF OCTET STRING OPTIONAL,
    ...,
    multipleCalls           BOOLEAN,
    maintainConnection      BOOLEAN,
    fastConnectRefused      NULL OPTIONAL
}
```

```
TransportAddress ::= CHOICE
{
     ipAddress SEQUENCE
     {
          ip             OCTET STRING (SIZE(4)),
          port           INTEGER(0..65535)
     },
     ipSourceRoute       SEQUENCE
     {
          ip             OCTET STRING (SIZE(4)),
          port           INTEGER(0..65535),
          route          SEQUENCE OF OCTET STRING (SIZE(4)),
          routing        CHOICE
          {
               strict    NULL,
               loose     NULL,
               ...
          },
          ...
     },
     ipxAddress     SEQUENCE
     {
          node           OCTET STRING (SIZE(6)),
          netnum         OCTET STRING (SIZE(4)),
          port           OCTET STRING (SIZE(2))
     },
     ip6Address     SEQUENCE
     {
          ip             OCTET STRING (SIZE(16)),
          port           INTEGER(0..65535),
          ...
     },
     netBios             OCTET STRING (SIZE(16)),
     nsap                OCTET STRING (SIZE(1..20)),
     nonStandardAddress NonStandardParameter,
     ...
}

Status-UUIE ::= SEQUENCE
{
     protocolIdentifier ProtocolIdentifier,
     callIdentifier      CallIdentifier,
     tokens              SEQUENCE OF ClearToken OPTIONAL,
     cryptoTokens        SEQUENCE OF CryptoH323Token OPTIONAL,
     ...
}

StatusInquiry-UUIE ::= SEQUENCE
{
     protocolIdentifier ProtocolIdentifier,
     callIdentifier      CallIdentifier,
     tokens              SEQUENCE OF ClearToken OPTIONAL,
     cryptoTokens        SEQUENCE OF CryptoH323Token OPTIONAL,
     ...
}

SetupAcknowledge-UUIE ::= SEQUENCE
{
     protocolIdentifier ProtocolIdentifier,
     callIdentifier      CallIdentifier,
     tokens              SEQUENCE OF ClearToken OPTIONAL,
     cryptoTokens        SEQUENCE OF CryptoH323Token OPTIONAL,
     ...
}
```

```
Notify-UUIE ::= SEQUENCE
{
    protocolIdentifier ProtocolIdentifier,
    callIdentifier     CallIdentifier,
    tokens             SEQUENCE OF ClearToken OPTIONAL,
    cryptoTokens       SEQUENCE OF CryptoH323Token OPTIONAL,
    ...
}

-- Beginning of common message elements section

EndpointType ::= SEQUENCE
{
    nonStandardData    NonStandardParameter OPTIONAL,
    vendor             VendorIdentifier OPTIONAL,
    gatekeeper         GatekeeperInfo OPTIONAL,
    gateway            GatewayInfo OPTIONAL,
    mcu                McuInfo OPTIONAL,  -- mc must be set as well
    terminal           TerminalInfo OPTIONAL,
    mc                 BOOLEAN,           -- shall not be set by itself
    undefinedNode      BOOLEAN,
    ...,
    set           BIT STRING (SIZE(32)) OPTIONAL,
                           -- shall not be used with mc, gatekeeper
                           -- code points for the various SET devices
                           -- are defined in the respective SET Annexes
    supportedTunnelledProtocols  SEQUENCE OF TunnelledProtocol OPTIONAL
                           -- list of supported tunnelled protocols
}

GatewayInfo ::= SEQUENCE
{
    protocol                SEQUENCE OF SupportedProtocols OPTIONAL,
    nonStandardData         NonStandardParameter OPTIONAL,
    ...
}

SupportedProtocols ::= CHOICE
{
    nonStandardData         NonStandardParameter,
    h310                    H310Caps,
    h320                    H320Caps,
    h321                    H321Caps,
    h322                    H322Caps,
    h323                    H323Caps,
    h324                    H324Caps,
    voice                   VoiceCaps,
    t120-only               T120OnlyCaps,
    ...,
    nonStandardProtocol     NonStandardProtocol,
    t38FaxAnnexbOnly        T38FaxAnnexbOnlyCaps,
    sip                     SIPCaps
}

H310Caps ::= SEQUENCE
{
    nonStandardData         NonStandardParameter OPTIONAL,
    ...,
    dataRatesSupported      SEQUENCE OF DataRate OPTIONAL,
    supportedPrefixes       SEQUENCE OF SupportedPrefix
}
```

```
H320Caps ::= SEQUENCE
{
    nonStandardData         NonStandardParameter OPTIONAL,
    ...,
    dataRatesSupported      SEQUENCE OF DataRate OPTIONAL,
    supportedPrefixes       SEQUENCE OF SupportedPrefix
}

H321Caps ::= SEQUENCE
{
    nonStandardData         NonStandardParameter OPTIONAL,
    ...,
    dataRatesSupported      SEQUENCE OF DataRate OPTIONAL,
    supportedPrefixes       SEQUENCE OF SupportedPrefix
}

H322Caps ::= SEQUENCE
{
    nonStandardData         NonStandardParameter OPTIONAL,
    ...,
    dataRatesSupported      SEQUENCE OF DataRate OPTIONAL,
    supportedPrefixes       SEQUENCE OF SupportedPrefix
}

H323Caps ::= SEQUENCE
{
    nonStandardData         NonStandardParameter OPTIONAL,
    ...,
    dataRatesSupported      SEQUENCE OF DataRate OPTIONAL,
    supportedPrefixes       SEQUENCE OF SupportedPrefix
}

H324Caps ::= SEQUENCE
{
    nonStandardData         NonStandardParameter OPTIONAL,
    ...,
    dataRatesSupported      SEQUENCE OF DataRate OPTIONAL,
    supportedPrefixes       SEQUENCE OF SupportedPrefix
}

VoiceCaps ::= SEQUENCE
{
    nonStandardData         NonStandardParameter OPTIONAL,
    ...,
    dataRatesSupported      SEQUENCE OF DataRate OPTIONAL,
    supportedPrefixes       SEQUENCE OF SupportedPrefix
}

T120OnlyCaps ::= SEQUENCE
{
    nonStandardData         NonStandardParameter OPTIONAL,
    ...,
    dataRatesSupported      SEQUENCE OF DataRate OPTIONAL,
    supportedPrefixes       SEQUENCE OF SupportedPrefix
}

NonStandardProtocol ::= SEQUENCE
{
    nonStandardData         NonStandardParameter OPTIONAL,
    dataRatesSupported      SEQUENCE OF DataRate OPTIONAL,
    supportedPrefixes       SEQUENCE OF SupportedPrefix,
    ...
}
```

```
T38FaxAnnexbOnlyCaps ::= SEQUENCE
{
    nonStandardData         NonStandardParameter OPTIONAL,
    dataRatesSupported      SEQUENCE OF DataRate OPTIONAL,
    supportedPrefixes       SEQUENCE OF SupportedPrefix,
    t38FaxProtocol          DataProtocolCapability,
    t38FaxProfile           T38FaxProfile,
    ...
}

SIPCaps ::= SEQUENCE
{
    nonStandardData         NonStandardParameter OPTIONAL,
    dataRatesSupported      SEQUENCE OF DataRate OPTIONAL,
    supportedPrefixes       SEQUENCE OF SupportedPrefix OPTIONAL,
    ...
}

McuInfo ::= SEQUENCE
{
    nonStandardData         NonStandardParameter OPTIONAL,
    ...,
    protocol                SEQUENCE OF SupportedProtocols OPTIONAL
}

TerminalInfo ::= SEQUENCE
{
    nonStandardData         NonStandardParameter OPTIONAL,
    ...
}

GatekeeperInfo ::= SEQUENCE
{
    nonStandardData         NonStandardParameter OPTIONAL,
    ...
}

VendorIdentifier ::= SEQUENCE
{
    vendor          H221NonStandard,
    productId       OCTET STRING (SIZE(1..256)) OPTIONAL,    -- per vendor
    versionId       OCTET STRING (SIZE(1..256)) OPTIONAL,    -- per product
    ...,
    enterpriseNumber    OBJECT IDENTIFIER OPTIONAL
}

H221NonStandard ::= SEQUENCE
{   t35CountryCode      INTEGER(0..255),
    t35Extension        INTEGER(0..255),
    manufacturerCode    INTEGER(0..65535),
    ...
}

TunnelledProtocol ::= SEQUENCE
{
    id CHOICE
    {
    tunnelledProtocolObjectID           OBJECT IDENTIFIER,
        tunnelledProtocolAlternateID    TunnelledProtocolAlternateIdentifier,
        ...
    },
    subIdentifier                       IA5String (SIZE (1..64)) OPTIONAL,
    ...
}
```

```
TunnelledProtocolAlternateIdentifier ::= SEQUENCE
{
     protocolType           IA5String (SIZE (1..64)),
     protocolVariant        IA5String (SIZE (1..64)) OPTIONAL,
     ...
}

NonStandardParameter ::= SEQUENCE
{
     nonStandardIdentifier  NonStandardIdentifier,
     data                   OCTET STRING
}

NonStandardIdentifier       ::= CHOICE
{
     object                 OBJECT IDENTIFIER,
     h221NonStandard        H221NonStandard,
     ...
}

AliasAddress ::= CHOICE
{
     dialledDigits IA5String (SIZE (1..128)) (FROM ("0123456789#*,")),
     h323-ID       BMPString (SIZE (1..256)), -- Basic ISO/IEC 10646-1 (Unicode)
     ...,
     url-ID        IA5String (SIZE(1..512)),  -- URL style address
     transportID   TransportAddress,
     email-ID      IA5String (SIZE(1..512)),  -- rfc822-compliant email address
     partyNumber   PartyNumber,
     mobileUIM     MobileUIM
}

AddressPattern ::= CHOICE
{
wildcard AliasAddress,
range         SEQUENCE
{
startOfRange  PartyNumber,
endOfRange    PartyNumber
},
...
}

PartyNumber ::= CHOICE
{
     e164Number                  PublicPartyNumber,
                                      -- the numbering plan is according to
                                      -- ITU-T Recs E.163 and E.164.
     dataPartyNumber             NumberDigits,
                                      -- not used, value reserved.
     telexPartyNumber            NumberDigits,
                                      -- not used, value reserved.
     privateNumber               PrivatePartyNumber,
                                      -- the numbering plan is according to
                                      -- ISO/IEC 11571.
     nationalStandardPartyNumber NumberDigits,
                                      -- not used, value reserved.
     ...
}
```

```
PublicPartyNumber ::= SEQUENCE
{
    publicTypeOfNumber      PublicTypeOfNumber,
    publicNumberDigits      NumberDigits
}

PrivatePartyNumber ::= SEQUENCE
{
    privateTypeOfNumber     PrivateTypeOfNumber,
    privateNumberDigits     NumberDigits
}

NumberDigits ::= IA5String (SIZE (1..128)) (FROM ("0123456789#*,"))

PublicTypeOfNumber ::= CHOICE
{
    unknown                 NULL,
                                      -- if used number digits carry prefix
                                      -- indicating type
                                      -- of number according to national
                                      -- recommendations.
    internationalNumber     NULL,
    nationalNumber          NULL,
    networkSpecificNumber   NULL,
                                      -- not used, value reserved
    subscriberNumber        NULL,
    abbreviatedNumber       NULL,
                                      -- valid only for called party number at
                                      -- the outgoing access, network substitutes
                                      -- appropriate number.
...
}

PrivateTypeOfNumber ::= CHOICE
{
    unknown                 NULL,
    level2RegionalNumber    NULL,
    level1RegionalNumber    NULL,
    pISNSpecificNumber      NULL,
    localNumber             NULL,
    abbreviatedNumber       NULL,
    ...
}

MobileUIM ::= CHOICE
{
    ansi-41-uim ANSI-41-UIM,     -- Americas standards Wireless Networks
    gsm-uim GSM-UIM,             -- European standards Wireless Networks
    ...
}

TBCD-STRING ::= IA5String (FROM ("0123456789#*abc"))
```

```
ANSI-41-UIM ::= SEQUENCE
{
    imsi                TBCD-STRING (SIZE (3..16)) OPTIONAL,
    min                 TBCD-STRING (SIZE (3..16)) OPTIONAL,
    mdn                 TBCD-STRING (SIZE (3..16)) OPTIONAL,
    msisdn              TBCD-STRING (SIZE (3..16)) OPTIONAL,
    esn                 TBCD-STRING (SIZE (16)) OPTIONAL,
    mscid               TBCD-STRING (SIZE (3..16)) OPTIONAL,
    system-id CHOICE
    {
        sid             TBCD-STRING (SIZE (1..4)),
        mid             TBCD-STRING (SIZE (1..4)),
        ...
    },
    systemMyTypeCode    OCTET STRING (SIZE (1)) OPTIONAL,
    systemAccessType    OCTET STRING (SIZE (1)) OPTIONAL,
    qualificationInformationCode OCTET STRING (SIZE (1)) OPTIONAL,
    sesn                TBCD-STRING (SIZE (16)) OPTIONAL,
    soc                 TBCD-STRING (SIZE (3..16)) OPTIONAL,
    ...
    -- IMSI refers to International Mobile Station Identification
    -- MIN refers to Mobile Identification Number
    -- MDN refers to Mobile Directory Number
    -- MSISDN refers to Mobile Station ISDN number
    -- ESN Refers to Electronic Serial Number
    -- MSCID refers to Mobile Switching Center number + Market ID or System ID
    -- SID refers to System Identification and MID refers to Market
    -- Identification
    -- SystemMyTypeCode refers to vendor identification number
    -- SystemAccessType refers to the system access type like power down
    -- registration or call
    -- origination or Short Message response etc.
    -- Qualification Information Code refers to the validity
    -- SESN Refers to SIM Electronic Serial Number for Security purposes of User
    -- Identification
    -- SOC refers to System Operator Code
}

GSM-UIM ::= SEQUENCE
{
    imsi                TBCD-STRING (SIZE (3..16)) OPTIONAL,
    tmsi                OCTET STRING (SIZE (1..4)) OPTIONAL,
    msisdn              TBCD-STRING (SIZE (3..16)) OPTIONAL,
    imei                TBCD-STRING (SIZE (15..16)) OPTIONAL,
    hplmn               TBCD-STRING (SIZE (1..4)) OPTIONAL,
    vplmn               TBCD-STRING (SIZE (1..4)) OPTIONAL,
    -- IMSI refers to International Mobile Station Identification
    -- MSISDN refers to Mobile Station ISDN number
    -- IMEI Refers to International Mobile Equipment Identification
    -- VPLMN or HPLMN refers to Visiting or Home Public Land Mobile Network
    -- number
    ...
}

ExtendedAliasAddress ::= SEQUENCE
{
    address                 AliasAddress,
    presentationIndicator   PresentationIndicator OPTIONAL,
    screeningIndicator      ScreeningIndicator OPTIONAL,
    ...
}
```

```
Endpoint ::= SEQUENCE
{
    nonStandardData             NonStandardParameter OPTIONAL,
    aliasAddress                SEQUENCE OF AliasAddress OPTIONAL,
    callSignalAddress           SEQUENCE OF TransportAddress OPTIONAL,
    rasAddress                  SEQUENCE OF TransportAddress OPTIONAL,
    endpointType                EndpointType OPTIONAL,
    tokens                      SEQUENCE OF ClearToken OPTIONAL,
    cryptoTokens                SEQUENCE OF CryptoH323Token OPTIONAL,
    priority                    INTEGER(0..127) OPTIONAL,
    remoteExtensionAddress      SEQUENCE OF AliasAddress OPTIONAL,
    destExtraCallInfo           SEQUENCE OF AliasAddress OPTIONAL,
    ...,
    alternateTransportAddresses AlternateTransportAddresses OPTIONAL,
    circuitInfo                 CircuitInfo   OPTIONAL,
    featureSet                  FeatureSet    OPTIONAL
}

AlternateTransportAddresses ::= SEQUENCE
{
    annexE              SEQUENCE OF TransportAddress OPTIONAL,
    ...,
    sctp                SEQUENCE OF TransportAddress OPTIONAL
}

UseSpecifiedTransport ::= CHOICE
{
    tcp                 NULL,
    annexE              NULL,
    ...,
    sctp                NULL
}

AlternateGK ::= SEQUENCE
{
    rasAddress              TransportAddress,
    gatekeeperIdentifier    GatekeeperIdentifier  OPTIONAL,
    needToRegister          BOOLEAN,
    priority                INTEGER (0..127),
    ...
}

AltGKInfo ::=SEQUENCE
{
    alternateGatekeeper     SEQUENCE OF AlternateGK,
    altGKisPermanent        BOOLEAN,
    ...
}

SecurityServiceMode ::= CHOICE
{
    nonStandard         NonStandardParameter,
    none                NULL,
    default             NULL,
    ...                 -- can be extended with other specific modes
}
```

```
SecurityCapabilities ::= SEQUENCE
{
     nonStandard             NonStandardParameter OPTIONAL,
     encryption              SecurityServiceMode,
     authenticaton           SecurityServiceMode,
     integrity               SecurityServiceMode,
     ...
}

SecurityErrors     ::=  CHOICE
{
     securityWrongSyncTime        NULL,     -- either time server
                                            -- problem or network delay
     securityReplay               NULL,     -- replay attack encountered
     securityWrongGeneralID       NULL,     -- wrong general ID
     securityWrongSendersID       NULL,     -- wrong senders ID
     securityIntegrityFailed      NULL,     -- integrity check failed
     securityWrongOID             NULL,     -- wrong token OIDs or crypto alg OIDs
     securityDHmismatch           NULL,     -- mismatch of DH parameters
     securityCertificateExpired   NULL,     -- certificate has expired
     securityCertificateDateInvalid   NULL,     -- certificate is not yet valid
     securityCertificateRevoked   NULL,     -- certificate was found revoked
     securityCertificateNotReadable   NULL,     -- decoding error
     securityCertificateSignatureInvalid   NULL,       -- wrong signature in the
                                                       -- certificate
     securityCertificateMissing   NULL,     -- no certificate available
     securityCertificateIncomplete NULL,    -- missing expected certificate
                                            -- extensions
     securityUnsupportedCertificateAlgOID  NULL, -- crypto algs not understood
     securityUnknownCA            NULL,     -- CA/root certificate could not
                                            -- be found
...
}

SecurityErrors2    ::=  CHOICE
{
     securityWrongSyncTime   NULL, -- either time server problem or network delay
     securityReplay          NULL, -- replay attack encountered
     securityWrongGeneralID  NULL, -- wrong general ID
     securityWrongSendersID  NULL, -- wrong senders ID
     securityIntegrityFailed NULL, -- integrity check failed
     securityWrongOID        NULL, -- wrong token OIDs or crypto alg OIDs
     ...
}

H245Security ::= CHOICE
{
     nonStandard             NonStandardParameter,
     noSecurity              NULL,
     tls                     SecurityCapabilities,
     ipsec                   SecurityCapabilities,
     ...
}
```

```
QseriesOptions ::= SEQUENCE
{
    q932Full      BOOLEAN, -- if true, indicates full support for Q.932
    q951Full      BOOLEAN, -- if true, indicates full support for Q.951
    q952Full      BOOLEAN, -- if true, indicates full support for Q.952
    q953Full      BOOLEAN, -- if true, indicates full support for Q.953
    q955Full      BOOLEAN, -- if true, indicates full support for Q.955
    q956Full      BOOLEAN, -- if true, indicates full support for Q.956
    q957Full      BOOLEAN, -- if true, indicates full support for Q.957
    q954Info      Q954Details,
    ...
}

Q954Details ::= SEQUENCE
{
    conferenceCalling       BOOLEAN,
    threePartyService       BOOLEAN,
    ...
}

GloballyUniqueID        ::=  OCTET STRING (SIZE(16))
ConferenceIdentifier    ::=  GloballyUniqueID
RequestSeqNum           ::=  INTEGER (1..65535)
GatekeeperIdentifier    ::=  BMPString (SIZE(1..128))
BandWidth               ::=  INTEGER (0..4294967295)   -- in 100s of bits
CallReferenceValue      ::=  INTEGER (0..65535)
EndpointIdentifier      ::=  BMPString (SIZE(1..128))
ProtocolIdentifier      ::=  OBJECT IDENTIFIER
TimeToLive              ::=  INTEGER (1..4294967295)   -- in seconds
H248PackagesDescriptor  ::=  OCTET STRING  -- This octet string contains ASN.1
                                           -- PER encoded H.248
                                           -- PackagesDescriptor

H248SignalsDescriptor   ::=  OCTET STRING  -- This octet string contains
                                           -- ASN.1 PER encoded H.248
                                           -- SignalsDescriptor.

FeatureDescriptor       ::=  GenericData

CallIdentifier ::= SEQUENCE
{
    guid          GloballyUniqueID,
    ...
}

EncryptIntAlg ::= CHOICE
{   -- core encryption algorithms for RAS message integrity
    nonStandard        NonStandardParameter,
    isoAlgorithm       OBJECT IDENTIFIER, -- defined in ISO/IEC 9979
    ...
}
NonIsoIntegrityMechanism ::= CHOICE
{   -- HMAC mechanism used, no truncation, tagging may be necessary!
    hMAC-MD5           NULL,
    hMAC-iso10118-2-s  EncryptIntAlg,     -- according to ISO/IEC 10118-2 using
                                          -- EncryptIntAlg as core block
                                          -- encryption algorithm (short MAC)
    hMAC-iso10118-2-l  EncryptIntAlg,     -- according to ISO/IEC 10118-2 using
                                          -- EncryptIntAlg as core block
                                          -- encryption algorithm (long MAC)
    hMAC-iso10118-3    OBJECT IDENTIFIER, -- according to ISO/IEC 10118-3 using
                                          -- OID as hash function (OID is SHA-1,
                                          -- RIPE-MD160,
                                          -- RIPE-MD128)
    ...
}

IntegrityMechanism ::= CHOICE
{   -- for RAS message integrity
    nonStandard    NonStandardParameter,
    digSig         NULL,               -- indicates to apply a digital signature
    iso9797        OBJECT IDENTIFIER, -- according to ISO/IEC 9797 using OID as
                                      -- core encryption algorithm (X-CBC MAC)
    nonIsoIM       NonIsoIntegrityMechanism,
    ...
}

ICV ::= SEQUENCE
{
    algorithmOID   OBJECT IDENTIFIER,     -- the algorithm used to compute the
                                          -- signature
    icv            BIT STRING             -- the computed cryptographic
                                          -- integrity check value or signature
}

FastStartToken ::= ClearToken (WITH COMPONENTS {..., timeStamp PRESENT, dhkey
PRESENT, generalID PRESENT
                                          -- set to "alias" -- })
EncodedFastStartToken ::= TYPE-IDENTIFIER.&Type (FastStartToken)
CryptoH323Token::= CHOICE
{
    cryptoEPPwdHash SEQUENCE
    {
        alias          AliasAddress,      -- alias of entity generating hash
        timeStamp      TimeStamp,         -- timestamp used in hash
        token          HASHED    { EncodedPwdCertToken -- generalID set to
                                                       -- "alias" -- }
    },
    cryptoGKPwdHash  SEQUENCE
    {
        gatekeeperId   GatekeeperIdentifier,   -- GatekeeperID of GK generating
                                               -- hash
        timeStamp      TimeStamp,              -- timestamp used in hash
        token          HASHED    { EncodedPwdCertToken  -- generalID set to
                                                        -- Gatekeeperid -- }
    },
    cryptoEPPwdEncr    ENCRYPTED { EncodedPwdCertToken -- generalID set to
                                                       -- Gatekeeperid --},
    cryptoGKPwdEncr    ENCRYPTED { EncodedPwdCertToken -- generalID set to
                                                       -- Gatekeeperid --},
    cryptoEPCert       SIGNED { EncodedPwdCertToken -- generalID set to
                                                    -- Gatekeeperid -- },
    cryptoGKCert       SIGNED { EncodedPwdCertToken -- generalID set to alias -- },
    cryptoFastStart    SIGNED { EncodedFastStartToken },
    nestedcryptoToken  CryptoToken,
    ...
}
```

```
DataRate ::= SEQUENCE
{
    nonStandardData         NonStandardParameter OPTIONAL,
    channelRate             BandWidth,
    channelMultiplier       INTEGER (1..256) OPTIONAL,
    ...
}

CallLinkage ::= SEQUENCE
{
    globalCallId            GloballyUniqueID  OPTIONAL,
    threadId                GloballyUniqueID  OPTIONAL,
    ...
}

SupportedPrefix ::= SEQUENCE
{
    nonStandardData         NonStandardParameter OPTIONAL,
    prefix                  AliasAddress,
    ...
}

CapacityReportingCapability ::= SEQUENCE
{
    canReportCallCapacity       BOOLEAN,
    ...
}

CapacityReportingSpecification ::= SEQUENCE
{
    when SEQUENCE
    {
        callStart     NULL OPTIONAL,
        callEnd       NULL OPTIONAL,
        ...
    },
    ...
}

CallCapacity ::= SEQUENCE
{
    maximumCallCapacity         CallCapacityInfo OPTIONAL,
    currentCallCapacity         CallCapacityInfo OPTIONAL,
    ...
}

CallCapacityInfo ::= SEQUENCE
{
    voiceGwCallsAvailable               SEQUENCE OF CallsAvailable OPTIONAL,
    h310GwCallsAvailable                SEQUENCE OF CallsAvailable OPTIONAL,
    h320GwCallsAvailable                SEQUENCE OF CallsAvailable OPTIONAL,
    h321GwCallsAvailable                SEQUENCE OF CallsAvailable OPTIONAL,
    h322GwCallsAvailable                SEQUENCE OF CallsAvailable OPTIONAL,
    h323GwCallsAvailable                SEQUENCE OF CallsAvailable OPTIONAL,
    h324GwCallsAvailable                SEQUENCE OF CallsAvailable OPTIONAL,
    t120OnlyGwCallsAvailable            SEQUENCE OF CallsAvailable OPTIONAL,
    t38FaxAnnexbOnlyGwCallsAvailable    SEQUENCE OF CallsAvailable OPTIONAL,
    terminalCallsAvailable              SEQUENCE OF CallsAvailable OPTIONAL,
    mcuCallsAvailable                   SEQUENCE OF CallsAvailable OPTIONAL,
    ...,
    sipGwCallsAvailable                 SEQUENCE OF CallsAvailable OPTIONAL
}
```

```
CallsAvailable ::= SEQUENCE
{
    calls           INTEGER (0..4294967295),
    group           IA5String (SIZE (1..128)) OPTIONAL,
    ...,
    carrier         CarrierInfo OPTIONAL
}

CircuitInfo ::= SEQUENCE
{
    sourceCircuitID             CircuitIdentifier OPTIONAL,
    destinationCircuitID        CircuitIdentifier OPTIONAL,
    genericData                 SEQUENCE OF GenericData OPTIONAL,
    ...
}

CircuitIdentifier ::= SEQUENCE
{
    cic             CicInfo OPTIONAL,  group     GroupID OPTIONAL,
    ...,
    carrier         CarrierInfo OPTIONAL
}

CicInfo ::= SEQUENCE
{
    cic             SEQUENCE OF OCTET STRING (SIZE (2..4)),
    pointCode       OCTET STRING (SIZE (2..5)),
    ...
}

GroupID ::= SEQUENCE
{
    member          SEQUENCE OF INTEGER (0..65535) OPTIONAL,
    group           IA5String (SIZE (1..128)),
    ...
}

CarrierInfo ::= SEQUENCE
{
    carrierIdentificationCode   OCTET STRING (SIZE (3..4)) OPTIONAL,
    carrierName                 IA5String (SIZE (1..128)) OPTIONAL,
    ...
}

ServiceControlDescriptor ::= CHOICE
{
    url                         IA5String (SIZE(0..512)),  -- indicates a URL-
                                                           -- referenced
                                                           -- protocol/resource
    signal                      H248SignalsDescriptor,
    nonStandard                 NonStandardParameter,
    callCreditServiceControl    CallCreditServiceControl,
    ...
}

ServiceControlSession ::= SEQUENCE
{
    sessionId       INTEGER (0..255),
    contents        ServiceControlDescriptor OPTIONAL,
    reason CHOICE
    {
        open            NULL,
        refresh         NULL,
        close           NULL,
        ...
    },
    ...
}

RasUsageInfoTypes ::= SEQUENCE
{
    nonStandardUsageTypes        SEQUENCE OF NonStandardParameter,
    startTime                    NULL OPTIONAL,
    endTime                      NULL OPTIONAL,
    terminationCause             NULL OPTIONAL,
    ...
}

RasUsageSpecification ::= SEQUENCE
{
    when SEQUENCE
    {
        start                    NULL OPTIONAL,
        end                      NULL OPTIONAL,
        inIrr                    NULL OPTIONAL,
        ...
    },
    callStartingPoint SEQUENCE
    {
        alerting            NULL OPTIONAL,
        connect             NULL OPTIONAL,
        ...
    } OPTIONAL,
    required                RasUsageInfoTypes,
    ...
}

RasUsageInformation ::= SEQUENCE
{
    nonStandardUsageFields       SEQUENCE OF NonStandardParameter,
    alertingTime                 TimeStamp OPTIONAL,
    connectTime                  TimeStamp OPTIONAL,
    endTime                      TimeStamp OPTIONAL,
    ...
}

CallTerminationCause ::= CHOICE
{
    releaseCompleteReason        ReleaseCompleteReason,
    releaseCompleteCauseIE       OCTET STRING (SIZE(2..32)),
    ...
}

BandwidthDetails ::= SEQUENCE
{
    sender              BOOLEAN,              -- TRUE=sender, FALSE=receiver
    multicast           BOOLEAN,              -- TRUE if stream is multicast
    bandwidth           BandWidth,            -- Bandwidth used for stream
    rtcpAddresses       TransportChannelInfo, -- RTCP addresses for media stream
    ...
}

CallCreditCapability ::= SEQUENCE
```

```
{
    canDisplayAmountString      BOOLEAN OPTIONAL,
    canEnforceDurationLimit     BOOLEAN OPTIONAL,
    ...
}

CallCreditServiceControl ::= SEQUENCE
{
    amountString            BMPString (SIZE (1..512)) OPTIONAL,   -- (Unicode)
    billingMode CHOICE
    {
        credit              NULL,
        debit               NULL,
        ...
    } OPTIONAL,
    callDurationLimit       INTEGER (1..4294967295) OPTIONAL,    -- in seconds
    enforceCallDurationLimit    BOOLEAN OPTIONAL,
    callStartingPoint CHOICE
    {
        alerting            NULL,
        connect             NULL,
        ...
    } OPTIONAL,
    ...
}

GenericData ::= SEQUENCE
{
    id                  GenericIdentifier,
    parameters          SEQUENCE (SIZE (1..512)) OF EnumeratedParameter OPTIONAL,
    ...
}

GenericIdentifier ::= CHOICE
{
    standard            INTEGER(0..16383,...),
    oid                 OBJECT IDENTIFIER,
    nonStandard         GloballyUniqueID,
    ...
}

EnumeratedParameter ::= SEQUENCE
{
    id                  GenericIdentifier,
    content             Content OPTIONAL,
    ...
}

Content ::= CHOICE
{
    raw                 OCTET STRING,
    text                IA5String,
    unicode             BMPString,
    bool                BOOLEAN,
    number8             INTEGER (0..255),
    number16            INTEGER (0..65535),
    number32            INTEGER (0..4294967295),
    id                  GenericIdentifier,
    alias               AliasAddress,
    transport           TransportAddress,
    compound            SEQUENCE (SIZE (1..512)) OF EnumeratedParameter,
    nested              SEQUENCE (SIZE (1..16)) OF GenericData,
    ...
}
```

```
FeatureSet ::= SEQUENCE
{
     replacementFeatureSet   BOOLEAN,
     neededFeatures          SEQUENCE OF FeatureDescriptor OPTIONAL,
     desiredFeatures         SEQUENCE OF FeatureDescriptor OPTIONAL,
     supportedFeatures       SEQUENCE OF FeatureDescriptor OPTIONAL,
     ...
}

TransportChannelInfo ::= SEQUENCE
{
     sendAddress             TransportAddress OPTIONAL,
     recvAddress             TransportAddress OPTIONAL,
     ...
}

RTPSession ::= SEQUENCE
{
     rtpAddress                   TransportChannelInfo,
     rtcpAddress                  TransportChannelInfo,
     cname                        PrintableString,
     ssrc                         INTEGER (1..4294967295),
     sessionId                    INTEGER (1..255),
     associatedSessionIds         SEQUENCE OF INTEGER (1..255),
     ...,
     multicast                    NULL OPTIONAL,
     bandwidth                    BandWidth OPTIONAL
}

RasMessage ::= CHOICE
{
     gatekeeperRequest            GatekeeperRequest,
     gatekeeperConfirm            GatekeeperConfirm,
     gatekeeperReject             GatekeeperReject,
     registrationRequest          RegistrationRequest,
     registrationConfirm          RegistrationConfirm,
     registrationReject           RegistrationReject,
     unregistrationRequest        UnregistrationRequest,
     unregistrationConfirm        UnregistrationConfirm,
     unregistrationReject         UnregistrationReject,
     admissionRequest             AdmissionRequest,
     admissionConfirm             AdmissionConfirm,
     admissionReject              AdmissionReject,
     bandwidthRequest             BandwidthRequest,
     bandwidthConfirm             BandwidthConfirm,
     bandwidthReject              BandwidthReject,
     disengageRequest             DisengageRequest,
     disengageConfirm             DisengageConfirm,
     disengageReject              DisengageReject,
     locationRequest              LocationRequest,
     locationConfirm              LocationConfirm,
     locationReject               LocationReject,
     infoRequest                  InfoRequest,
     infoRequestResponse          InfoRequestResponse,
     nonStandardMessage           NonStandardMessage,
     unknownMessageResponse       UnknownMessageResponse,
     ...,
     requestInProgress            RequestInProgress,
     resourcesAvailableIndicate   ResourcesAvailableIndicate,
     resourcesAvailableConfirm    ResourcesAvailableConfirm,
     infoRequestAck              InfoRequestAck,
     infoRequestNak              InfoRequestNak,
     serviceControlIndication    ServiceControlIndication,
     serviceControlResponse      ServiceControlResponse,
     admissionConfirmSequence    SEQUENCE OF AdmissionConfirm
}

GatekeeperRequest ::= SEQUENCE --(GRQ)
{

     requestSeqNum               RequestSeqNum,
     protocolIdentifier          ProtocolIdentifier,
     nonStandardData             NonStandardParameter OPTIONAL,
     rasAddress                  TransportAddress,
     endpointType                EndpointType,
     gatekeeperIdentifier        GatekeeperIdentifier OPTIONAL,
     callServices                QseriesOptions  OPTIONAL,
     endpointAlias               SEQUENCE OF AliasAddress OPTIONAL,
     ...,
     alternateEndpoints          SEQUENCE OF Endpoint OPTIONAL,
     tokens                      SEQUENCE OF ClearToken OPTIONAL,
     cryptoTokens                SEQUENCE OF CryptoH323Token OPTIONAL,
     authenticationCapability    SEQUENCE OF AuthenticationMechanism OPTIONAL,
     algorithmOIDs               SEQUENCE OF OBJECT IDENTIFIER OPTIONAL,
     integrity                   SEQUENCE OF IntegrityMechanism OPTIONAL,
     integrityCheckValue         ICV OPTIONAL,
     supportsAltGK               NULL OPTIONAL,
     featureSet                  FeatureSet OPTIONAL,
     genericData                 SEQUENCE OF GenericData OPTIONAL
}

GatekeeperConfirm ::= SEQUENCE --(GCF)
{
     requestSeqNum               RequestSeqNum,
     protocolIdentifier          ProtocolIdentifier,
     nonStandardData             NonStandardParameter OPTIONAL,
     gatekeeperIdentifier        GatekeeperIdentifier  OPTIONAL,
     rasAddress                  TransportAddress,
     ...,
     alternateGatekeeper         SEQUENCE OF AlternateGK OPTIONAL,
     authenticationMode          AuthenticationMechanism OPTIONAL,
     tokens                      SEQUENCE OF ClearToken OPTIONAL,
     cryptoTokens                SEQUENCE OF CryptoH323Token OPTIONAL,
     algorithmOID                OBJECT IDENTIFIER OPTIONAL,
     integrity                   SEQUENCE OF IntegrityMechanism OPTIONAL,
     integrityCheckValue         ICV OPTIONAL,
     featureSet                  FeatureSet OPTIONAL,
     genericData                 SEQUENCE OF GenericData OPTIONAL
}

GatekeeperReject ::= SEQUENCE --(GRJ)
{
     requestSeqNum               RequestSeqNum,
     protocolIdentifier          ProtocolIdentifier,
     nonStandardData             NonStandardParameter OPTIONAL,
     gatekeeperIdentifier        GatekeeperIdentifier  OPTIONAL,
     rejectReason                GatekeeperRejectReason,
     ...,
     altGKInfo                   AltGKInfo OPTIONAL,
     tokens                      SEQUENCE OF ClearToken OPTIONAL,
     cryptoTokens                SEQUENCE OF CryptoH323Token OPTIONAL,
     integrityCheckValue         ICV OPTIONAL,
     featureSet                  FeatureSet OPTIONAL,
     genericData                 SEQUENCE OF GenericData OPTIONAL
}

GatekeeperRejectReason ::= CHOICE
{
     resourceUnavailable         NULL,
     terminalExcluded            NULL,     -- permission failure, not a resource
                                           -- failure
     invalidRevision             NULL,
     undefinedReason             NULL,
     ...,
     securityDenial              NULL,
     genericDataReason           NULL,
     neededFeatureNotSupported   NULL,
securityError               SecurityErrors}

RegistrationRequest ::= SEQUENCE --(RRQ)
{
     requestSeqNum               RequestSeqNum,
     protocolIdentifier          ProtocolIdentifier,
     nonStandardData             NonStandardParameter OPTIONAL,
     discoveryComplete           BOOLEAN,
     callSignalAddress           SEQUENCE OF TransportAddress,
     rasAddress                  SEQUENCE OF TransportAddress,
     terminalType                EndpointType,
     terminalAlias               SEQUENCE OF AliasAddress OPTIONAL,
     gatekeeperIdentifier        GatekeeperIdentifier  OPTIONAL,
     endpointVendor              VendorIdentifier,
     ...,
     alternateEndpoints          SEQUENCE OF Endpoint OPTIONAL,
     timeToLive                  TimeToLive OPTIONAL,
     tokens                      SEQUENCE OF ClearToken OPTIONAL,
     cryptoTokens                SEQUENCE OF CryptoH323Token OPTIONAL,
     integrityCheckValue         ICV OPTIONAL,
     keepAlive                   BOOLEAN,
     endpointIdentifier          EndpointIdentifier OPTIONAL,
     willSupplyUUIEs             BOOLEAN,
     maintainConnection          BOOLEAN,
     alternateTransportAddresses AlternateTransportAddresses OPTIONAL,
     additiveRegistration        NULL OPTIONAL,
     terminalAliasPattern        SEQUENCE OF AddressPattern OPTIONAL,
     supportsAltGK               NULL OPTIONAL,
     usageReportingCapability    RasUsageInfoTypes OPTIONAL,
     multipleCalls               BOOLEAN OPTIONAL,
     supportedH248Packages       SEQUENCE OF H248PackagesDescriptor OPTIONAL,
     callCreditCapability        CallCreditCapability OPTIONAL,
     capacityReportingCapability CapacityReportingCapability OPTIONAL,
     capacity                    CallCapacity OPTIONAL,
     featureSet                  FeatureSet OPTIONAL,
     genericData                 SEQUENCE OF GenericData OPTIONAL,
     restart                     NULL     OPTIONAL,
     supportsACFSequences        NULL     OPTIONAL
}

RegistrationConfirm ::= SEQUENCE --(RCF)
{
     requestSeqNum                        RequestSeqNum,
     protocolIdentifier                   ProtocolIdentifier,
     nonStandardData                      NonStandardParameter OPTIONAL,
     callSignalAddress                    SEQUENCE OF TransportAddress,
     terminalAlias                        SEQUENCE OF AliasAddress OPTIONAL,
     gatekeeperIdentifier                  GatekeeperIdentifier  OPTIONAL,
     endpointIdentifier                    EndpointIdentifier,
     ...,
     alternateGatekeeper                   SEQUENCE OF AlternateGK OPTIONAL,
     timeToLive                            TimeToLive OPTIONAL,
     tokens                                SEQUENCE OF ClearToken OPTIONAL,
     cryptoTokens                          SEQUENCE OF CryptoH323Token OPTIONAL,
     integrityCheckValue                   ICV OPTIONAL,
     willRespondToIRR                      BOOLEAN,
     preGrantedARQ                         SEQUENCE
     {
          makeCall                         BOOLEAN,
          useGKCallSignalAddressToMakeCall BOOLEAN,
          answerCall                       BOOLEAN,
          useGKCallSignalAddressToAnswer   BOOLEAN,
          ...,
          irrFrequencyInCall      INTEGER (1..65535) OPTIONAL, -- in seconds;
                                                              -- not present
                                                              -- if GK does
                                                              -- not want IRRs
          totalBandwidthRestriction   BandWidth OPTIONAL,      -- total limit
                                                              -- for all
                                                              -- concurrent calls
          alternateTransportAddresses      AlternateTransportAddresses OPTIONAL,
          useSpecifiedTransport            UseSpecifiedTransport OPTIONAL
     } OPTIONAL,
     maintainConnection               BOOLEAN,
     serviceControl                   SEQUENCE OF ServiceControlSession OPTIONAL,
     supportsAdditiveRegistration     NULL OPTIONAL,
     terminalAliasPattern             SEQUENCE OF AddressPattern OPTIONAL,
     supportedPrefixes                SEQUENCE OF SupportedPrefix OPTIONAL,
     usageSpec                        SEQUENCE OF RasUsageSpecification OPTIONAL,
     featureServerAlias               AliasAddress OPTIONAL,
     capacityReportingSpec            CapacityReportingSpecification OPTIONAL,
     featureSet                       FeatureSet OPTIONAL,
     genericData                      SEQUENCE OF GenericData OPTIONAL
}

RegistrationReject ::= SEQUENCE --(RRJ)
{
     requestSeqNum               RequestSeqNum,
     protocolIdentifier          ProtocolIdentifier,
     nonStandardData             NonStandardParameter OPTIONAL,
     rejectReason                RegistrationRejectReason,
     gatekeeperIdentifier        GatekeeperIdentifier  OPTIONAL,
     ...,
     altGKInfo                   AltGKInfo OPTIONAL,
     tokens                      SEQUENCE OF ClearToken OPTIONAL,
     cryptoTokens                SEQUENCE OF CryptoH323Token OPTIONAL,
     integrityCheckValue         ICV OPTIONAL,
     featureSet                  FeatureSet OPTIONAL,
     genericData                 SEQUENCE OF GenericData OPTIONAL
}

RegistrationRejectReason ::= CHOICE
{
     discoveryRequired           NULL,
     invalidRevision             NULL,
     invalidCallSignalAddress    NULL,
     invalidRASAddress           NULL,     -- supplied address is invalid
     duplicateAlias              SEQUENCE OF AliasAddress,
                                           -- alias registered to another
                                           -- endpoint
     invalidTerminalType         NULL,
     undefinedReason             NULL,
     transportNotSupported       NULL,     -- one or more of the transports
     ...,
     transportQOSNotSupported    NULL,     -- endpoint QoS not supported
     resourceUnavailable         NULL,     -- gatekeeper resources exhausted
     invalidAlias                NULL,     -- alias not consistent with
                                           -- gatekeeper rules
     securityDenial              NULL,
     fullRegistrationRequired    NULL,     -- registration permission has expired
     additiveRegistrationNotSupported NULL,
     invalidTerminalAliases           SEQUENCE
     {
          terminalAlias          SEQUENCE OF AliasAddress OPTIONAL,
          terminalAliasPattern   SEQUENCE OF AddressPattern OPTIONAL,
          supportedPrefixes      SEQUENCE OF SupportedPrefix OPTIONAL,
          ...
     },
     genericDataReason           NULL,
     neededFeatureNotSupported   NULL,
     securityError               SecurityErrors
}

UnregistrationRequest ::= SEQUENCE --(URQ)
{
     requestSeqNum            RequestSeqNum,
     callSignalAddress        SEQUENCE OF TransportAddress,
     endpointAlias            SEQUENCE OF AliasAddress OPTIONAL,
     nonStandardData          NonStandardParameter OPTIONAL,
     endpointIdentifier       EndpointIdentifier OPTIONAL,
     ...,
     alternateEndpoints       SEQUENCE OF Endpoint OPTIONAL,
     gatekeeperIdentifier     GatekeeperIdentifier OPTIONAL,
     tokens                   SEQUENCE OF ClearToken OPTIONAL,
     cryptoTokens             SEQUENCE OF CryptoH323Token OPTIONAL,
     integrityCheckValue      ICV OPTIONAL,
     reason                   UnregRequestReason OPTIONAL,
     endpointAliasPattern     SEQUENCE OF AddressPattern OPTIONAL,
     supportedPrefixes        SEQUENCE OF SupportedPrefix OPTIONAL,
     alternateGatekeeper      SEQUENCE OF AlternateGK OPTIONAL,
     genericData              SEQUENCE OF GenericData OPTIONAL
}

UnregRequestReason ::= CHOICE
{
     reregistrationRequired  NULL,
     ttlExpired              NULL,
     securityDenial          NULL,
     undefinedReason         NULL,
     ...,
     maintenance             NULL,
     securityError           SecurityErrors2
}

UnregistrationConfirm ::= SEQUENCE --(UCF)
{
     requestSeqNum            RequestSeqNum,
     nonStandardData          NonStandardParameter OPTIONAL,
     ...,
     tokens                   SEQUENCE OF ClearToken OPTIONAL,
     cryptoTokens             SEQUENCE OF CryptoH323Token OPTIONAL,
     integrityCheckValue      ICV OPTIONAL,
     genericData              SEQUENCE OF GenericData OPTIONAL
}
```

```
UnregistrationReject ::= SEQUENCE --(URJ)
{

     requestSeqNum           RequestSeqNum,
     rejectReason            UnregRejectReason,
     nonStandardData         NonStandardParameter OPTIONAL,
     ...,
     altGKInfo               AltGKInfo OPTIONAL,
     tokens                  SEQUENCE OF ClearToken OPTIONAL,
     cryptoTokens            SEQUENCE OF CryptoH323Token OPTIONAL,
     integrityCheckValue     ICV OPTIONAL,
     genericData             SEQUENCE OF GenericData OPTIONAL
}

UnregRejectReason ::= CHOICE
{
     notCurrentlyRegistered  NULL,
     callInProgress          NULL,
     undefinedReason         NULL,
     ...,
     permissionDenied        NULL,     -- requesting user not allowed to
                                       -- unregister specified user
     securityDenial          NULL,
     securityError           SecurityErrors2
}

AdmissionRequest ::= SEQUENCE --(ARQ)
{
     requestSeqNum                RequestSeqNum,
     callType                     CallType,
     callModel                    CallModel OPTIONAL,
     endpointIdentifier           EndpointIdentifier,
     destinationInfo              SEQUENCE OF AliasAddress OPTIONAL,
     destCallSignalAddress        TransportAddress OPTIONAL,
     destExtraCallInfo            SEQUENCE OF AliasAddress OPTIONAL,
     srcInfo                      SEQUENCE OF AliasAddress,
     srcCallSignalAddress         TransportAddress OPTIONAL,
     bandWidth                    BandWidth,
     callReferenceValue           CallReferenceValue,
     nonStandardData              NonStandardParameter OPTIONAL,
     callServices                 QseriesOptions  OPTIONAL,
     conferenceID                 ConferenceIdentifier,
     activeMC                     BOOLEAN,
     answerCall                   BOOLEAN,  -- answering a call
     ...,
     canMapAlias                  BOOLEAN,  -- can handle alias address
     callIdentifier               CallIdentifier,
     srcAlternatives              SEQUENCE OF Endpoint OPTIONAL,
     destAlternatives             SEQUENCE OF Endpoint OPTIONAL,
     gatekeeperIdentifier         GatekeeperIdentifier OPTIONAL,
     tokens                       SEQUENCE OF ClearToken OPTIONAL,
     cryptoTokens                 SEQUENCE OF CryptoH323Token OPTIONAL,
     integrityCheckValue          ICV OPTIONAL,
     transportQOS                 TransportQOS OPTIONAL,
     willSupplyUUIEs              BOOLEAN,
     callLinkage                  CallLinkage OPTIONAL,
     gatewayDataRate              DataRate OPTIONAL,
     capacity                     CallCapacity OPTIONAL,
     circuitInfo                  CircuitInfo OPTIONAL,
     desiredProtocols             SEQUENCE OF SupportedProtocols OPTIONAL,
    desiredTunnelledProtocol    TunnelledProtocol OPTIONAL,
    featureSet                  FeatureSet OPTIONAL,
    genericData                 SEQUENCE OF GenericData OPTIONAL,
    canMapSrcAlias              BOOLEAN
}

CallType ::= CHOICE
{
    pointToPoint        NULL,         -- Point-to-point
    oneToN              NULL,         -- no interaction (FFS)
    nToOne              NULL,         -- no interaction (FFS)
    nToN                NULL,         -- interactive (multipoint)
    ...
}

CallModel ::= CHOICE
{
    direct              NULL,
    gatekeeperRouted    NULL,
    ...
}

TransportQOS ::= CHOICE
{
    endpointControlled      NULL,
    gatekeeperControlled    NULL,
    noControl               NULL,
    ...
}

AdmissionConfirm ::= SEQUENCE --(ACF)
{
    requestSeqNum               RequestSeqNum,
    bandWidth                   BandWidth,
    callModel                   CallModel,
    destCallSignalAddress       TransportAddress,
    irrFrequency                INTEGER (1..65535) OPTIONAL,
    nonStandardData             NonStandardParameter OPTIONAL,
    ...,
    destinationInfo             SEQUENCE OF AliasAddress OPTIONAL,
    destExtraCallInfo           SEQUENCE OF AliasAddress OPTIONAL,
    destinationType             EndpointType OPTIONAL,
    remoteExtensionAddress      SEQUENCE OF AliasAddress OPTIONAL,
    alternateEndpoints          SEQUENCE OF Endpoint OPTIONAL,
    tokens                      SEQUENCE OF ClearToken OPTIONAL,
    cryptoTokens                SEQUENCE OF CryptoH323Token OPTIONAL,
    integrityCheckValue         ICV OPTIONAL,
    transportQOS                TransportQOS OPTIONAL,
    willRespondToIRR            BOOLEAN,
    uuiesRequested              UUIEsRequested,
    language                    SEQUENCE OF IA5String (SIZE (1..32)) OPTIONAL,
    alternateTransportAddresses AlternateTransportAddresses OPTIONAL,
    useSpecifiedTransport       UseSpecifiedTransport OPTIONAL,
    circuitInfo                 CircuitInfo OPTIONAL,
    usageSpec                   SEQUENCE OF RasUsageSpecification OPTIONAL,
    supportedProtocols          SEQUENCE OF SupportedProtocols OPTIONAL,
    serviceControl              SEQUENCE OF ServiceControlSession OPTIONAL,
    multipleCalls               BOOLEAN OPTIONAL,
    featureSet                  FeatureSet OPTIONAL,
    genericData                 SEQUENCE OF GenericData OPTIONAL,
    modifiedSrcInfo             SEQUENCE OF AliasAddress OPTIONAL
}
```

```
UUIEsRequested ::= SEQUENCE
{
     setup                    BOOLEAN,
     callProceeding           BOOLEAN,
     connect                  BOOLEAN,
     alerting                 BOOLEAN,
     information              BOOLEAN,
     releaseComplete          BOOLEAN,
     facility                 BOOLEAN,
     progress                 BOOLEAN,
     empty                    BOOLEAN,
     ...,
     status                   BOOLEAN,
     statusInquiry            BOOLEAN,
     setupAcknowledge         BOOLEAN,
     notify                   BOOLEAN
}

AdmissionReject ::= SEQUENCE --(ARJ)
{
     requestSeqNum            RequestSeqNum,
     rejectReason             AdmissionRejectReason,
     nonStandardData          NonStandardParameter OPTIONAL,
     ...,
     altGKInfo                AltGKInfo OPTIONAL,
     tokens                   SEQUENCE OF ClearToken OPTIONAL,
     cryptoTokens             SEQUENCE OF CryptoH323Token OPTIONAL,
     callSignalAddress        SEQUENCE OF TransportAddress OPTIONAL,
     integrityCheckValue      ICV OPTIONAL,
     serviceControl           SEQUENCE OF ServiceControlSession OPTIONAL,
     featureSet               FeatureSet OPTIONAL,
     genericData              SEQUENCE OF GenericData OPTIONAL
}

AdmissionRejectReason ::= CHOICE
{
     calledPartyNotRegistered     NULL,     -- cannot translate address
     invalidPermission            NULL,     -- permission has expired
     requestDenied                NULL,     -- no bandwidth available
     undefinedReason              NULL,
     callerNotRegistered          NULL,
     routeCallToGatekeeper        NULL,
     invalidEndpointIdentifier    NULL,
     resourceUnavailable          NULL,
     ...,
     securityDenial               NULL,
     qosControlNotSupported       NULL,
     incompleteAddress            NULL,
     aliasesInconsistent          NULL,     -- multiple aliases in request
                                            -- identify distinct people
     routeCallToSCN               SEQUENCE OF PartyNumber,
     exceedsCallCapacity          NULL,     -- destination does not have the
                                            -- capacity for this call
     collectDestination           NULL,
     collectPIN                   NULL,
     genericDataReason            NULL,
     neededFeatureNotSupported    NULL,
     securityError                SecurityErrors2,
     securityDHmismatch           NULL,     -- mismatch of DH parameters
     noRouteToDestination         NULL,     -- destination unreachable
     unallocatedNumber            NULL      -- destination number unassigned
}
```

```
BandwidthRequest ::= SEQUENCE --(BRQ)
{
     requestSeqNum            RequestSeqNum,
     endpointIdentifier       EndpointIdentifier,
     conferenceID             ConferenceIdentifier,
     callReferenceValue       CallReferenceValue,
     callType                 CallType OPTIONAL,
     bandWidth                BandWidth,
     nonStandardData          NonStandardParameter OPTIONAL,
     ...,
     callIdentifier           CallIdentifier,
     gatekeeperIdentifier     GatekeeperIdentifier OPTIONAL,
     tokens                   SEQUENCE OF ClearToken OPTIONAL,
     cryptoTokens             SEQUENCE OF CryptoH323Token OPTIONAL,
     integrityCheckValue      ICV OPTIONAL,
     answeredCall             BOOLEAN,
     callLinkage              CallLinkage OPTIONAL,
     capacity                 CallCapacity OPTIONAL,
     usageInformation         RasUsageInformation OPTIONAL,
     bandwidthDetails         SEQUENCE OF BandwidthDetails OPTIONAL,
     genericData              SEQUENCE OF GenericData OPTIONAL
}

BandwidthConfirm ::= SEQUENCE --(BCF)
{
     requestSeqNum                 RequestSeqNum,
     bandWidth                     BandWidth,
     nonStandardData               NonStandardParameter OPTIONAL,
     ...,
     tokens                        SEQUENCE OF ClearToken OPTIONAL,
     cryptoTokens                  SEQUENCE OF CryptoH323Token OPTIONAL,
     integrityCheckValue           ICV OPTIONAL,
     capacity                      CallCapacity OPTIONAL,
     genericData                   SEQUENCE OF GenericData OPTIONAL
}

BandwidthReject ::= SEQUENCE --(BRJ)
{
     requestSeqNum                 RequestSeqNum,
     rejectReason                  BandRejectReason,
     allowedBandWidth              BandWidth,
     nonStandardData               NonStandardParameter OPTIONAL,
     ...,
     altGKInfo                     AltGKInfo OPTIONAL,
     tokens                        SEQUENCE OF ClearToken OPTIONAL,
     cryptoTokens                  SEQUENCE OF CryptoH323Token OPTIONAL,
     integrityCheckValue           ICV OPTIONAL,
     genericData                   SEQUENCE OF GenericData OPTIONAL
}

BandRejectReason ::= CHOICE
{
     notBound                      NULL,     -- discovery permission has aged
     invalidConferenceID           NULL,     -- possible revision
     invalidPermission             NULL,     -- true permission violation
     insufficientResources         NULL,
     invalidRevision               NULL,
     undefinedReason               NULL,
     ...,
     securityDenial                NULL,
     securityError                 SecurityErrors2}
```

```
LocationRequest ::= SEQUENCE --(LRQ)
{
     requestSeqNum                RequestSeqNum,
     endpointIdentifier           EndpointIdentifier OPTIONAL,
     destinationInfo              SEQUENCE OF AliasAddress,
     nonStandardData              NonStandardParameter OPTIONAL,
     replyAddress                 TransportAddress,
     ...,
     sourceInfo                   SEQUENCE OF AliasAddress OPTIONAL,
     canMapAlias                  BOOLEAN, -- can handle alias address
     gatekeeperIdentifier         GatekeeperIdentifier OPTIONAL,
     tokens                       SEQUENCE OF ClearToken OPTIONAL,
     cryptoTokens                 SEQUENCE OF CryptoH323Token OPTIONAL,
     integrityCheckValue          ICV OPTIONAL,
     desiredProtocols             SEQUENCE OF SupportedProtocols OPTIONAL,
     desiredTunnelledProtocol     TunnelledProtocol OPTIONAL,
     featureSet                   FeatureSet OPTIONAL,
     genericData                  SEQUENCE OF GenericData OPTIONAL,
     hopCount                     INTEGER (1..255) OPTIONAL,
     circuitInfo                  CircuitInfo OPTIONAL,
     callIdentifier               CallIdentifier OPTIONAL,
     bandWidth                    BandWidth OPTIONAL,
     sourceEndpointInfo           SEQUENCE OF AliasAddress OPTIONAL,
     canMapSrcAlias               BOOLEAN
}

LocationConfirm ::= SEQUENCE --(LCF)
{
     requestSeqNum                RequestSeqNum,
     callSignalAddress            TransportAddress,
     rasAddress                   TransportAddress,
     nonStandardData              NonStandardParameter OPTIONAL,
     ...,
     destinationInfo              SEQUENCE OF AliasAddress OPTIONAL,
     destExtraCallInfo            SEQUENCE OF AliasAddress OPTIONAL,
     destinationType              EndpointType OPTIONAL,
     remoteExtensionAddress       SEQUENCE OF AliasAddress OPTIONAL,
     alternateEndpoints           SEQUENCE OF Endpoint OPTIONAL,
     tokens                       SEQUENCE OF ClearToken OPTIONAL,
     cryptoTokens                 SEQUENCE OF CryptoH323Token OPTIONAL,
     integrityCheckValue          ICV OPTIONAL,
     alternateTransportAddresses  AlternateTransportAddresses OPTIONAL,
     supportedProtocols           SEQUENCE OF SupportedProtocols OPTIONAL,
     multipleCalls                BOOLEAN OPTIONAL,
     featureSet                   FeatureSet OPTIONAL,
     genericData                  SEQUENCE OF GenericData OPTIONAL,
     circuitInfo                  CircuitInfo OPTIONAL,
     serviceControl               SEQUENCE OF ServiceControlSession OPTIONAL,
     modifiedSrcInfo              SEQUENCE OF AliasAddress OPTIONAL,
     bandWidth                    BandWidth OPTIONAL
}

LocationReject ::= SEQUENCE --(LRJ)
{
     requestSeqNum           RequestSeqNum,
     rejectReason            LocationRejectReason,
     nonStandardData         NonStandardParameter OPTIONAL,
     ...,
     altGKInfo               AltGKInfo OPTIONAL,
     tokens                  SEQUENCE OF ClearToken OPTIONAL,
     cryptoTokens            SEQUENCE OF CryptoH323Token OPTIONAL,
     integrityCheckValue     ICV OPTIONAL,
     featureSet              FeatureSet OPTIONAL,
     genericData             SEQUENCE OF GenericData OPTIONAL,
     serviceControl          SEQUENCE OF ServiceControlSession OPTIONAL
}

LocationRejectReason ::= CHOICE
{
     notRegistered           NULL,
     invalidPermission       NULL,     -- exclusion by administrator or feature
     requestDenied           NULL,     -- cannot find location
     undefinedReason         NULL,
     ...,
     securityDenial              NULL,
     aliasesInconsistent         NULL,     -- multiple aliases in request
                                           -- identify distinct people
     routeCalltoSCN              SEQUENCE OF PartyNumber,
     resourceUnavailable         NULL,
     genericDataReason           NULL,
     neededFeatureNotSupported   NULL,
     hopCountExceeded            NULL,
     incompleteAddress           NULL,
     securityError               SecurityErrors2,
     securityDHmismatch          NULL,     -- mismatch of DH parameters
     noRouteToDestination        NULL,     -- destination unreachable
     unallocatedNumber           NULL      -- destination number unassigned
}

DisengageRequest ::= SEQUENCE --(DRQ)
{
     requestSeqNum           RequestSeqNum,
     endpointIdentifier      EndpointIdentifier,
     conferenceID            ConferenceIdentifier,
     callReferenceValue      CallReferenceValue,
     disengageReason         DisengageReason,
     nonStandardData         NonStandardParameter OPTIONAL,
     ...,
     callIdentifier          CallIdentifier,
     gatekeeperIdentifier    GatekeeperIdentifier OPTIONAL,
     tokens                  SEQUENCE OF ClearToken OPTIONAL,
     cryptoTokens            SEQUENCE OF CryptoH323Token OPTIONAL,
     integrityCheckValue     ICV OPTIONAL,
     answeredCall            BOOLEAN,
     callLinkage             CallLinkage OPTIONAL,
     capacity                CallCapacity OPTIONAL,
     circuitInfo             CircuitInfo OPTIONAL,
     usageInformation        RasUsageInformation OPTIONAL,
     terminationCause        CallTerminationCause OPTIONAL,
     serviceControl          SEQUENCE OF ServiceControlSession OPTIONAL,
     genericData             SEQUENCE OF GenericData OPTIONAL
}

DisengageReason ::= CHOICE
{
     forcedDrop              NULL,     -- gatekeeper is forcing the drop
     normalDrop              NULL,     -- associated with normal drop
     undefinedReason         NULL,
     ...
}

DisengageConfirm ::= SEQUENCE --(DCF)
{
     requestSeqNum           RequestSeqNum,
     nonStandardData         NonStandardParameter OPTIONAL,
     ...,
     tokens                  SEQUENCE OF ClearToken OPTIONAL,
     cryptoTokens            SEQUENCE OF CryptoH323Token OPTIONAL,
     integrityCheckValue     ICV OPTIONAL,
     capacity                CallCapacity OPTIONAL,
     circuitInfo             CircuitInfo OPTIONAL,
     usageInformation        RasUsageInformation OPTIONAL,
     genericData             SEQUENCE OF GenericData OPTIONAL
}

DisengageReject ::= SEQUENCE --(DRJ)
{
     requestSeqNum           RequestSeqNum,
     rejectReason            DisengageRejectReason,
     nonStandardData         NonStandardParameter OPTIONAL,
     ...,
     altGKInfo               AltGKInfo OPTIONAL,
     tokens                  SEQUENCE OF ClearToken OPTIONAL,
     cryptoTokens            SEQUENCE OF CryptoH323Token OPTIONAL,
     integrityCheckValue     ICV OPTIONAL,
     genericData             SEQUENCE OF GenericData OPTIONAL
}

DisengageRejectReason ::= CHOICE
{
     notRegistered           NULL,     -- not registered with gatekeeper
     requestToDropOther      NULL,     -- cannot request drop for others
     ...,
     securityDenial          NULL,
     securityError           SecurityErrors2
}

InfoRequest ::= SEQUENCE --(IRQ)
{
     requestSeqNum           RequestSeqNum,
     callReferenceValue      CallReferenceValue,
     nonStandardData         NonStandardParameter OPTIONAL,
     replyAddress            TransportAddress OPTIONAL,
     ...,
     callIdentifier          CallIdentifier,
     tokens                  SEQUENCE OF ClearToken OPTIONAL,
     cryptoTokens            SEQUENCE OF CryptoH323Token OPTIONAL,
     integrityCheckValue     ICV OPTIONAL,
     uuiesRequested          UUIEsRequested OPTIONAL,
     callLinkage             CallLinkage OPTIONAL,
     usageInfoRequested          RasUsageInfoTypes OPTIONAL,
     segmentedResponseSupported  NULL OPTIONAL,
     nextSegmentRequested        INTEGER (0..65535) OPTIONAL,
     capacityInfoRequested       NULL OPTIONAL,
     genericData                 SEQUENCE OF GenericData OPTIONAL
}

InfoRequestResponse ::= SEQUENCE --(IRR)
{

     nonStandardData         NonStandardParameter OPTIONAL,
     requestSeqNum           RequestSeqNum,
     endpointType            EndpointType,
     endpointIdentifier      EndpointIdentifier,
     rasAddress              TransportAddress,
     callSignalAddress       SEQUENCE OF TransportAddress,
     endpointAlias           SEQUENCE OF AliasAddress OPTIONAL,
     perCallInfo             SEQUENCE OF SEQUENCE
     {
         nonStandardData         NonStandardParameter OPTIONAL,
         callReferenceValue      CallReferenceValue,
         conferenceID            ConferenceIdentifier,
         originator               BOOLEAN OPTIONAL,
         audio                    SEQUENCE OF RTPSession OPTIONAL,
         video                    SEQUENCE OF RTPSession OPTIONAL,
         data                     SEQUENCE OF TransportChannelInfo OPTIONAL,
         h245                     TransportChannelInfo,
         callSignalling           TransportChannelInfo,
         callType                 CallType,
         bandWidth                BandWidth,
         callModel                CallModel,
         ...,
         callIdentifier           CallIdentifier,
         tokens                   SEQUENCE OF ClearToken OPTIONAL,
         cryptoTokens             SEQUENCE OF CryptoH323Token OPTIONAL,
         substituteConfIDs        SEQUENCE OF ConferenceIdentifier,
         pdu                      SEQUENCE OF SEQUENCE
         {
             h323pdu       H323-UU-PDU,
             sent      BOOLEAN       -- TRUE is sent, FALSE is received
         } OPTIONAL,
         callLinkage              CallLinkage OPTIONAL,
         usageInformation         RasUsageInformation OPTIONAL,
         circuitInfo              CircuitInfo OPTIONAL
     } OPTIONAL,
     ...,
     tokens                  SEQUENCE OF ClearToken OPTIONAL,
     cryptoTokens            SEQUENCE OF CryptoH323Token OPTIONAL,
     integrityCheckValue     ICV OPTIONAL,
     needResponse            BOOLEAN,
     capacity                CallCapacity OPTIONAL,
     irrStatus               InfoRequestResponseStatus OPTIONAL,
     unsolicited             BOOLEAN,
     genericData             SEQUENCE OF GenericData OPTIONAL
}

InfoRequestResponseStatus ::= CHOICE
{
     complete           NULL,
     incomplete         NULL,
     segment            INTEGER (0..65535),
     invalidCall        NULL,
     ...
}

InfoRequestAck ::= SEQUENCE --(IACK)
{
     requestSeqNum                RequestSeqNum,
     nonStandardData              NonStandardParameter OPTIONAL,
     tokens                       SEQUENCE OF ClearToken OPTIONAL,
     cryptoTokens                 SEQUENCE OF CryptoH323Token OPTIONAL,
     integrityCheckValue          ICV OPTIONAL,
     ...
}

InfoRequestNak ::= SEQUENCE --(INAK)
{
     requestSeqNum                RequestSeqNum,
     nonStandardData              NonStandardParameter OPTIONAL,
     nakReason                    InfoRequestNakReason,
     altGKInfo                    AltGKInfo OPTIONAL,
     tokens                       SEQUENCE OF ClearToken OPTIONAL,
     cryptoTokens                 SEQUENCE OF CryptoH323Token OPTIONAL,
     integrityCheckValue          ICV OPTIONAL,
     ...
}
```

```
InfoRequestNakReason ::= CHOICE
{
    notRegistered       NULL,     -- not registered with gatekeeper
    securityDenial      NULL,
    undefinedReason     NULL,
    ...,
    securityError       SecurityErrors2

}

NonStandardMessage ::= SEQUENCE
{
    requestSeqNum           RequestSeqNum,
    nonStandardData         NonStandardParameter,
    ...,
    tokens                  SEQUENCE OF ClearToken OPTIONAL,
    cryptoTokens            SEQUENCE OF CryptoH323Token OPTIONAL,
    integrityCheckValue     ICV OPTIONAL,
    featureSet              FeatureSet OPTIONAL,
    genericData             SEQUENCE OF GenericData OPTIONAL
}

UnknownMessageResponse ::= SEQUENCE   -- (XRS)
{
    requestSeqNum           RequestSeqNum,
    ...,
    tokens                  SEQUENCE OF ClearToken OPTIONAL,
    cryptoTokens            SEQUENCE OF CryptoH323Token OPTIONAL,
    integrityCheckValue     ICV OPTIONAL,
    messageNotUnderstood    OCTET STRING
}

RequestInProgress ::= SEQUENCE        -- (RIP)
{
    requestSeqNum           RequestSeqNum,
    nonStandardData         NonStandardParameter     OPTIONAL,
    tokens                  SEQUENCE OF ClearToken OPTIONAL,
    cryptoTokens            SEQUENCE OF CryptoH323Token OPTIONAL,
    integrityCheckValue     ICV OPTIONAL,
    delay                   INTEGER(1..65535),
    ...
}

ResourcesAvailableIndicate ::= SEQUENCE --(RAI)
{
    requestSeqNum           RequestSeqNum,
    protocolIdentifier      ProtocolIdentifier,
    nonStandardData         NonStandardParameter OPTIONAL,
    endpointIdentifier      EndpointIdentifier,
    protocols               SEQUENCE OF SupportedProtocols,
    almostOutOfResources    BOOLEAN,
    tokens                  SEQUENCE OF ClearToken OPTIONAL,
    cryptoTokens            SEQUENCE OF CryptoH323Token OPTIONAL,
    integrityCheckValue     ICV OPTIONAL,
    ...,
    capacity                CallCapacity OPTIONAL,
    genericData             SEQUENCE OF GenericData OPTIONAL
}
```

```
ResourcesAvailableConfirm ::= SEQUENCE --(RAC)
{
    requestSeqNum           RequestSeqNum,
    protocolIdentifier      ProtocolIdentifier,
    nonStandardData         NonStandardParameter OPTIONAL,
    tokens                  SEQUENCE OF ClearToken OPTIONAL,
    cryptoTokens            SEQUENCE OF CryptoH323Token OPTIONAL,
    integrityCheckValue     ICV OPTIONAL,
    ...,
    genericData             SEQUENCE OF GenericData OPTIONAL
}

ServiceControlIndication ::= SEQUENCE --(SCI)
{
    requestSeqNum           RequestSeqNum,
    nonStandardData         NonStandardParameter OPTIONAL,
    serviceControl          SEQUENCE OF ServiceControlSession,
    endpointIdentifier      EndpointIdentifier OPTIONAL,
    callSpecific SEQUENCE
    {
        callIdentifier      CallIdentifier,
        conferenceID        ConferenceIdentifier,
        answeredCall        BOOLEAN,
        ...
    } OPTIONAL,
    tokens                      SEQUENCE OF ClearToken OPTIONAL,
    cryptoTokens                SEQUENCE OF CryptoH323Token OPTIONAL,
    integrityCheckValue         ICV OPTIONAL,
    featureSet                  FeatureSet OPTIONAL,
    genericData                 SEQUENCE OF GenericData OPTIONAL,
    ...
}

ServiceControlResponse ::= SEQUENCE --(SCR)
{
    requestSeqNum                   RequestSeqNum,
    result                          CHOICE
    {
        started                     NULL,
        failed                      NULL,
        stopped                     NULL,
        notAvailable                NULL,
        neededFeatureNotSupported   NULL,
        ...
    } OPTIONAL,
    nonStandardData             NonStandardParameter OPTIONAL,
    tokens                      SEQUENCE OF ClearToken OPTIONAL,
    cryptoTokens                SEQUENCE OF CryptoH323Token OPTIONAL,
    integrityCheckValue         ICV OPTIONAL,
    featureSet                  FeatureSet OPTIONAL,
    genericData                 SEQUENCE OF GenericData OPTIONAL,
    ...
}


END       -- of ASN.1
```

Annex I

H.263+ video packetization

IETF RFC 2429 specifies the RTP payload format for H.263 video bitstreams that contain the new "H.263+" features adopted in version 2 (dated 1998) of ITU-T Rec. H.263 (includes the features using PLUSTYPE or Annex I/H.263 through Annex T/H.263).

The ability to support the H.263 payload format of RFC 2190 as specified in Annex E is required for H.263 bitstreams which do not use the new version 2 features of ITU-T Rec. H.263, because this support is needed for compatibility with prior implementations. However, the new payload format specified in RFC 2429 should be used even for bitstreams which do not contain the new version 2 features, provided the newer payload format is within the capabilities of the receiving terminals.

Appendix I

RTP/RTCP algorithms

The referenced informative material may be found in the following proposed Internet Standard:

– SCHULZRINNE (H.), CASNER (S.), FREDERICK (R.) and JACOBSON (V.): RFC 3550, RTP: A Transport Protocol for Real-Time Applications, *Internet Engineering Task Force*, 2003.

Appendix II

RTP profile

The referenced informative material may be found in the following proposed Internet Standard:

– SCHULZRINNE (H.), CASNER (S.): RFC 3551, RTP Profile for Audio and Video Conferences with Minimal Control, *Internet Engineering Task Force*, 2003.

Appendix III

H.261 packetization

The referenced informative material may be found in the following proposed Internet Standard:

– TURLETTI (T.), HUITEMA (C.): RFC 2032, RTP Payload Format for H.261 Video Streams, *Internet Engineering Task Force*, 1996.

Appendix IV

H.225.0 operation on different packet-based network protocol stacks

This appendix provides additional details concerning the operation of H.225.0 on various actual packet-based network protocol stacks. Packet-based networks used in this Recommendation shall provide both reliable and unreliable modes of operation, including a means to distinguish packet boundaries.

IV.1 TCP/IP/UDP

Note that UDP can fragment and reassemble large video packets, but that failure to perform MB packetization may lead to the loss of an entire GOB.

IP multicast should be used for GRQ distribution as opposed to media access layer broadcast.

Unreliable delivery applications	**Call signalling and H.245 channel**
UDP	TPKT ___ ___ TCP
IP	
Link Layer	
Physical Layer	

A TPKT is a packet format as defined in IETF RFC 1006. It is used to delimit individual messages (PDUs) within the TCP stream, which itself provides a continuous stream of octets without explicit boundaries. A TPKT consists of a one-octet version number field, followed by a one-octet reserved field, followed by a two-octet length field, followed by the actual data. The version number field shall contain the value "3", the reserved field shall contain the value "0". The length field shall contain the length of the entire packet including the version number, the reserved and the length fields as a 16-bit big-endian word.

IV.1.1 Discovering the gatekeeper

IV.1.1.1 Discovery using multicast address or well-known port

Following the gatekeeper discovery and registration procedures described in clause 7/H.323, endpoints should use the following multicast address or well known port when attempting to discover the gatekeeper as appropriate for their network configuration:

– UDP Address for multicast communication with gatekeepers: 224.0.1.41

– UDP port for multicast communication with gatekeepers: 1718

– UDP port for unicast RAS communication where no "other agreement" exists: 1719

Note that "other agreement" may include registration of an endpoint with a gatekeeper.

Note that implementations should pay attention to the scope of the multicast so as not to flood the Internet with discovery messages.

Assuming a gatekeeper has an IP address for example of 134.134.12.1, the following signalling may occur:

- LRQ or GRQ arrives at 134.134.12.1: port 1719;
- LRQ or GRQ arrives at 134.134.12.1: port 1718 (note that this may occur with v1 GKs);
- LRQ or GRQ arrives at 224.0.1.41: port 1718.

The gatekeeper may transmit an LRQ to the following addresses:

- 224.0.1.41: port 1718 (multicast to all GKs);
- X.X.X.X: port 1719 (to a specific GK).

Port 1719 should only be used when a request is sent unicast. This allows the receiver to know whether it should send a reject (xRJ) to the sender (it should in all cases).

Port 1718 should only be used when a request is sent multicast. The receiver should respond with the appropriate response, depending on the message. For LRQ no reject required, the receiver does not reply for multicast requests. For GRQ, a directed GRJ should be sent to the source of the GRQ.

IV.1.1.2 Discovery using DNS (informative)

IV.1.1.2.1 A URL for gatekeepers

As a first step, note that a gatekeeper is identified by a transport address and a gatekeeperIdentifier, which is a string. A gatekeeper is a particular resource on the Internet, so it is reasonable to specify it in a Uniform Resource Locator (URL). The protocol spoken by the gatekeeper is RAS, so the URL for a gatekeeper could be given by:

ras://gkID@domainname

gkID is the gatekeeperIdentifier, and domainname is a DNS domain name which identifies the gatekeeper's domain. Note that this is not necessarily a Fully Qualified Domain Name (FQDN) with an A-record: it is not required that this domain name has a physical transport interface with an IP number recorded in the DNS. If it is a FQDN, however, it is reasonable to insist that its IP number is that of the gatekeeper to which the URL refers. In this case, it is allowed to add an optional port number to the URL:

ras://gkID@domainname:port_no.

If no port number is given, then the well known value of 1719 is taken as a default.

The more interesting case is when this is not an FQDN, and then the domain name does not refer to a transport address listed in the DNS. The domain name then can refer to a pure "gatekeeper zone of authority". The next clause explains how to find the gatekeeper in this case.

IV.1.1.2.2 Finding the URL

The URL does not solve the problem of locating the gatekeeper, it just gives a standard format for the information to find. The problem is how to produce a transport address and gatekeeperIdentifier for RAS signalling given the domain name of a gatekeeper.

If the gatekeeper has an IETF RFC 822-compliant identifier, it is easy to extract a domain name from the IETF RFC 822-compliant identifier of a gatekeeper. In fact, it may be convenient to give IETF RFC 822-compliant identifiers to endpoints, and then to stipulate that the domain name part of the identifier refers to the gatekeeper domain.

IV.1.1.2.2.1 The SRV resource record query

The first solution uses the fact that the gatekeeper is basically a system service, and the transport address of a named system service can be extracted from DNS by using a query for a new type of DNS Resource Record, called SRV (for "service location record"). Given a domain name, an

SRV record query will be made for the transport address of the RAS service for that domain. The domain name itself, or one returned in the SRV response, is used as the gatekeeperIdentifier. The SRV record and its use are defined in IETF RFC 2782.

IV.1.1.2.2.2 The TXT record query

All current DNS implementations support the TXT resource record. Basically, this is some free text that can be returned for each domain name. It is possible to store many TXT resources for a single domain. The standard stipulates that all TXT records will be returned when a query is made for them.

It is possible to use TXT queries if the SRV queries fail. Assume the same convention for extracting a domain name that was suggested above. Either IETF RFC 822 compliant strings (email "-like" names) or IETF RFC 1768 compliant strings (URLs) can be used for gatekeeperIdentifiers. In either case the domain name is used to make a DNS TXT query for the domain name. The returned resource records are lines of free text, and the terminal will then look for lines in the response of the form:

ras [< gk id>@]<domain name >[:<portno>] [<priority>]

The **<gk id>** field is an optional gatekeeper ID which is separate from the domain name. If this field is missing, then the domain itself is assumed to be the gatekeeper ID.

The **<domain name>** field can be either the name of the A-record which contains the gatekeeper's IP address, or a raw IP address in dotted form. The domain name need not be fully qualified; if it is not, the subdomain in which the TXT record was found should be appended to it to form the fully qualified A-record name.

The optional **[:<portno>]** can be used to specify a port number other than the standard RAS port.

The optional **[<priority>]** field specifies the order in which the listed gatekeepers should be accessed for discovery or LRQ queries if there is more than one RAS TXT record. Lower numbers have higher priority.

Note that this format, if the **<gk id>** field is missing, assumes that the gatekeeper IDs are in fact legal domain names. However, if it is necessary for a single host to support multiple logical gatekeepers, each with a separate ID, the format will support this. This is because separate A-records can contain the same IP address.

White spaces are used as delimiters between **ras** and **gk id** if present or **domain name**, and between **portno** and **priority**. White spaces consist of any number of spaces or tabs.

Examples of valid gatekeeper TXT records:

– ras gk1

– ras gk1.company.com

– ras gk1:1500 3

– ras 172.11.22.33:1500 2

The client parses the returned lines, and from them obtains the transport address of the gatekeeper within that domain to which it can send RAS messages.

Since DNS requires a server to return all TXT records associated with a domain name, the client can filter out and process only those records which are useful to it. It also allows DNS to return an ordered list of gatekeepers which can serve as alternatives and back-ups as defined in ITU-T Rec. H.323.

Note that the server returned in such a query might be an actual transport address in dotted decimal notation, or it could be an FQDN which itself requires an A-record query in DNS to determine the transport address. The advantage of using an FQDN is the usual hiding of actual IP numbers. The advantage of using IP numbers is that a second DNS query is avoided, thus speeding up this pre-call setup time.

IV.1.1.2.3 Gatekeeper processing of email-IDs during ARQ and LRQ

When the **destinationInfo** field of an ARQ or LRQ message contains an **email-ID** alias address, the gatekeeper should first check its registration database for the alias. If it cannot be resolved, the gatekeeper should parse the alias to recover its domain portion. If no domain is given, the gatekeeper may generate a default domain. The domain is then used to locate one or more gatekeepers, using the procedures in IV.1.1.2.2. The gatekeeper may then query all gatekeepers thus found with an LRQ/LCF/LRJ message exchange.

Note that more than one gatekeeper may have corresponding TXT records in a single DNS domain. Consequently, a single DNS domain can "contain" more than one H.323 zone. Therefore, even if a gatekeeper cannot resolve an email-ID whose domain portion is one of its default domains, it may still query other zones in the same DNS domain.

If the gatekeeper is presented with an unregistered alias which is an **h323-id** and the ID can be interpreted as a legal user portion of an IETF RFC 822 name, the gatekeeper may interpret the alias as if it were an email-ID in its default domain and attempt to locate the alias in some other gatekeeper. Similarly, an email-ID from an incoming LRQ may be stripped of its domain name by the gatekeeper so that it may be located as an h323-ID.

IV.1.2 Endpoint-to-endpoint communications

Endpoints which wish to receive calls from endpoints outside the zone of their gatekeeper should use the following port for the Call Signalling channel:

– Endpoint TCPCall Signalling Port 1720

While it is permitted to use dynamic values for these ports to allow multiple endpoints in a single device, it must be understood that this will prevent interoperation with endpoints outside the zone of the gatekeeper except via a gateway in the zone.

IV.2 SPX/IPX

Note that since there is no network reassembly of large packets, the use of MB fragmentation is essential.

Unreliable delivery applications	H.245 channel call signalling channel
PXP	SPX
IPX	
Link Layer	
Physical Layer	

IV.2.1 Discovering the gatekeeper

In IPX terminology, a "socket" is the equivalent of a "port" in IP and a "TSAP Identifier" in this Recommendation and in ITU-T Rec. H.323.

On IPX-based networks, the gatekeepers should advertise the "gatekeeper service type" defined below to allow endpoints to locate them on a network. Likewise, endpoints should query for the "gatekeeper service type" to find the location of the nearest gatekeeper.

– Gatekeeper Service Type FFS.

NOTE – The service type is referred to as the SAP socket in some IPX documentation.

IV.2.2 Endpoint-to-endpoint communication

Endpoints which wish to receive calls from endpoints outside the zone of their gatekeeper should use the following sockets for Call signalling.

– Endpoint IPX Call Signalling Port FFS.

While it is permitted to use dynamic values for these sockets to allow multiple endpoints in a single device, it must be understood that this will prevent interoperation with endpoints outside the zone of the gatekeeper except via a gateway in the zone.

IV.3 SCTP

The protocol stack of H.323 over SCTP looks as follows:

Unreliable delivery applications	**Call signalling with tunnelled call control**
UDP	SCTP
IP	
Link Layer	
Physical Layer	

Each H.225.0 call signalling message shall be transferred in separate SCTP DATA chunk. No headers shall be added (i.e., no TPKT). Ordered delivery shall be specified.

IV.3.1 Streams

All messages of the same call shall use same SCTP stream. Implementation may use different streams for different calls.

IV.3.2 Payload protocol identifiers

SCTP may be used with undefined payload protocol identifier (0) or with 13, which is the number assigned to H.323 by IANA.

Appendix V

ASN.1 usage in this Recommendation

This appendix lists the ASN.1 conventions that have been used in this Recommendation. Futures revisions of this Recommendation should use only these constructs. Additional ASN.1 constructs will only be considered in exceptional circumstances.

V.1 Tagging

All tags within this Recommendation are AUTOMATIC TAGS.

V.2 Types

The following types may occur in the ASN.1 definitions of this Recommendation.

BIT STRING
BMPString
BOOLEAN
CHOICE
GeneralString
IA5String
INTEGER
NULL
NumericString
OBJECT IDENTIFIER
OCTET STRING
SEQUENCE
SEQUENCE OF
SET
SET OF

V.3 Constraints and ranges

This Recommendation uses size constraints ("SIZE") for strings, SET OF and SEQUENCE OF, value range constraints for integers, and permitted alphabets ("FROM").

V.4 Extensibility

This Recommendation uses the extension marker (ellipsis "...").

Appendix VI

H.225.0 identifiers of tunnelled signalling protocols

This Recommendation supports the tunnelling of non-H.323 call signalling protocols, as described in 10.4/H.323. The Annex M/H.323 series of Recommendations (M.1/H.323, M.2/H.323, etc.) defines tunnelling for specific protocols. A tunnelled protocol in this Recommendation is identified by information in the **TunnelledProtocol** ASN.1 structure defined in 7.6 and in Annex H. This appendix provides a list of **TunnelledProtocol** identifiers that have been allocated to specific tunnelled protocols.

Tunnelled protocols that are defined as of this Recommendation are shown in Tables VI.1 and VI.2. Note that tunnelling is not restricted to the protocols listed in these tables.

Table VI.1/H.225.0 – Tunnelled protocols identified by tunnelledProtocolObjectID

Tunnelling specification	Protocol specification	tunnelledProtocolObjectID	subIdentifier
M.1/H.323	ISO/IEC 11572 and 11582	{iso (1) identified-organization (3) icd-ecma (0012) private-isdn-signalling-domain (9)}	(None)
M.2/H.323	ITU-T Rec. Q.763 (1988)	{itu-t (0) recommendation (0) q (17) 763}	"1988"
M.2/H.323	ITU-T Rec. Q.763 (1993)	{itu-t (0) recommendation (0) q (17) 763}	"1993"

Table VI.2/H.225.0 – Tunnelled protocols identified by TunnelledProtocolAlternateIdentifier

Tunnelling specification	Protocol specification	protocolType	protocolVariant	subIdentifier
M.2/H.323	ANSI T1.113-1988	"isup"	"ANSI T1.113-1988"	"1988"
M.2/H.323	ETS 300 121	"isup"	"ETS 300 121"	"121"
M.2/H.323	ETS 300 356	"isup"	"ETS 300 356"	"356"
M.2/H.323	BELLCORE GR-317	"isup"	"BELLCORE GR-317"	"317"
M.2/H.323	JT-Q761-4(1987-1992)	"isup"	"JT-Q761-4(1987-1992)"	"87"
M.2/H.323	JT-Q761-4(1993)	"isup"	"JT-Q761-4(1993)"	"93"

ITU-T RECOMMENDATIONS SERIES

Series A	Organization of the work of the 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	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
Series Z	Programming languages

Printed in Switzerland
Geneva, 2004