INTERNATIONAL TELECOMMUNICATION UNION

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

H.225.0
Annex G
(05/99)

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

Annex G: Communication between administrative domains

ITU-T Recommendation H.225.0 – Annex G
Superseded by a more recent version

(Previously CCITT Recommendation)

ITU-T H-SERIES RECOMMENDATIONS

AUDIOVISUAL AND MULTIMEDIA SYSTEMS

Characteristics of transmission channels used for other than telephone purposes	H.10–H.19
Use of telephone-type circuits for voice-frequency telegraphy	H.20–H.29
Telephone circuits or cables used for various types of telegraph transmission or simultaneous transmission	H.30–H.39
Telephone-type circuits used for facsimile telegraphy	H.40–H.49
Characteristics of data signals	H.50–H.99
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

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

ITU-T RECOMMENDATION H.225.0

CALL SIGNALLING PROTOCOLS AND MEDIA STREAM PACKETIZATION FOR PACKET-BASED MULTIMEDIA COMMUNICATION SYSTEMS

ANNEX G

Communication between administrative domains

Summary

This annex 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.

Source

Annex G to ITU-T Recommendation H.225.0 was prepared by ITU-T Study Group 16 (1997-2000) and was approved under the WTSC Resolution No. 1 procedure on 27 May 1999.

FOREWORD

ITU (International Telecommunication Union) is the United Nations Specialized Agency in the field of telecommunications. The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of the ITU. The 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 Conference (WTSC), which meets every four years, establishes the topics for study by the ITU-T Study Groups which, in their turn, produce Recommendations on these topics.

The approval of Recommendations by the Members of the ITU-T is covered by the procedure laid down in WTSC Resolution No. 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 term *recognized operating agency (ROA)* includes any individual, company, corporation or governmental organization that operates a public correspondence service. The terms *Administration*, *ROA* and *public correspondence* are defined in the *Constitution of the ITU (Geneva, 1992)*.

INTELLECTUAL PROPERTY RIGHTS

The ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. The 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, the 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
G.1	Scope	1
G.2	Definitions	2
G.3	Abbreviations	3
G.4	References	3
G.5	System Models	3
G.5.1	Hierarchical	4
G.5.2	Distributed or Full Mesh	4
G.5.3	Clearing House	5
G.5.4	Aggregation Point	5
G.5.5	Overlapping Administrative Domains	6
G.6	Addressing Conventions	6
G.7	Operation	6
G.7.1	Address Templates and Descriptors	6
G.7.2	Discovery of a Border Element or Set of Border Elements	9
G.7.3	Resolution Procedures	9
G.7.4	Usage Information Exchange	10
G.8	Protocol	10
G.8.1	Security Considerations	10
G.8.2	Message Definitions	11
G.9	Signalling Examples	25
G.9.1	Distributed or Full Mesh	26
G.9.2	Clearing House	29

Recommendation H.225.0

CALL SIGNALLING PROTOCOLS AND MEDIA STREAM PACKETIZATION FOR PACKET-BASED MULTIMEDIA COMMUNICATION SYSTEMS

ANNEX G

Communication between administrative domains

(Geneva, 1999)

G.1 Scope

It is expected that the overall H.323 network will consist of smaller subsets of equipment organized in a manner such as by administrative domains. Because of the potentially large numbers of H.323 equipment 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 provide many of the needs of communication between administrative domains, it is neither complete nor efficient for this purpose.

This annex describes methods to allow address resolution, access authorization and usage reporting 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. A border 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 border elements to exchange information regarding the addresses each administrative domain 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 this figure, the administrative domains are part of a global packet network without edges. Note that this figure is not an explicit definition of an H.323 system architecture, but is meant to illustrate signalling reference points.

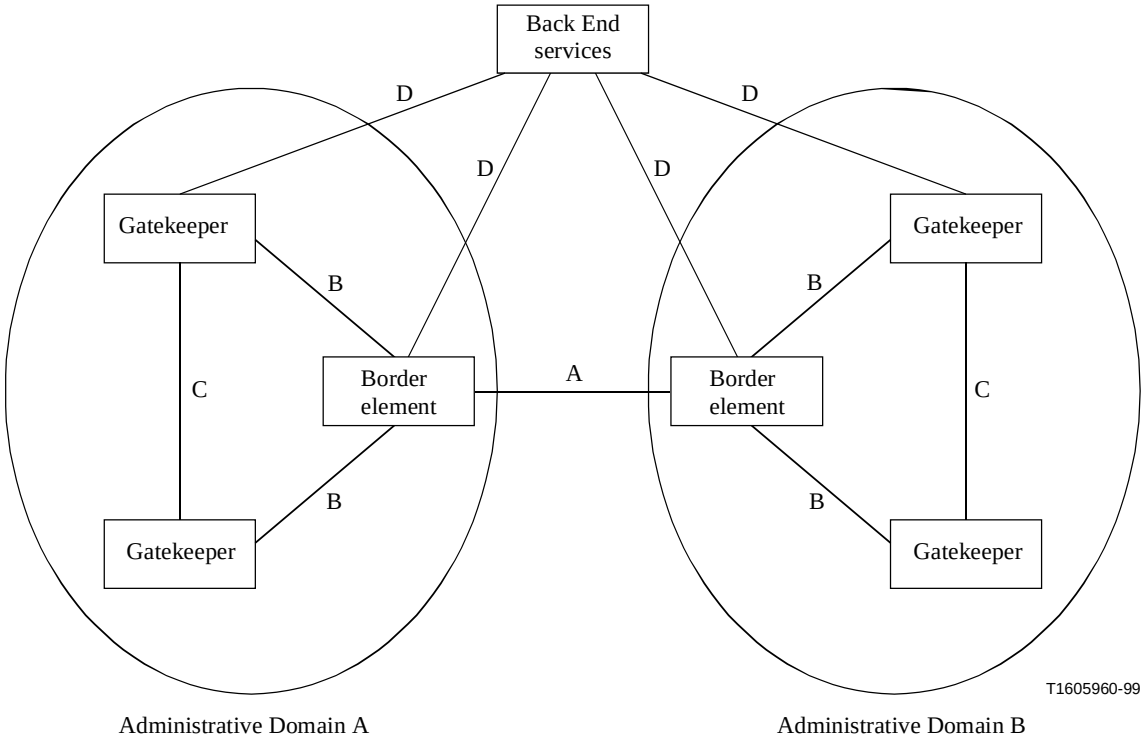

Figure G.1/H.225.0 – System Reference Points

The figure indicates the following reference points:

A – between border elements.

B – between border element and gatekeepers.

C – between gatekeepers.

D – between H.323 elements and back-end services (not in the scope of this annex).

Reference point A is the focus of Annex G. Use of the protocol described in Annex G for communication between gatekeepers within an administrative domain is for further study. Reference point B is considered for further study since it is currently assumed that the border element will be colocated with some other H.323 element.

Subclause G.9, Signalling Examples, provides some signalling examples which may aid understanding.

G.2 Definitions

This Recommendation 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 Border element: The border element is a functional element which 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. A border element communicates

with other border elements using the protocol specified in this annex. In addition, a border element may, depending on implementation, communicate with other entities within its administrative domain. This element may exist in combination with other H.323 elements, for example a combination of border element, gatekeeper, and gateway. An administrative domain may contain any number of border elements.

G.2.4 Clearing House: A service (possibly in the form of a border element) which can provide resolution for all addresses (i.e. a type of aggregation point).

G.3 Abbreviations

This Recommendation uses the following abbreviations:

AD	Administrative domain
BE	Border element
CH	Clearing house
DST	Daylight saving time
EP	Endpoint
GK	Gatekeeper
GW	Gateway
T	Terminal

G.4 References

[1] ITU-T Recommendation H.225.0 (1998), *Call signalling protocols and media stream packetization for packet-based multimedia communication systems*.

[2] ITU-T Recommendation H.235 (1998), *Security and encryption for H-series (H.323 and other H.245-based multimedia terminals)*.

[3] ITU-T Recommendation H.323 (1998), *Packet based multimedia communications systems*.

[4] ITU-T Recommendation X.680 (1997) | ISO/IEC 8824-1:1998, *Information technology – Abstract Syntax Notation One (ASN.1): Specification of basic notation*.

[5] ITU-T Recommendation X.680 (1997)/Amd.1 (1999) | ISO/IEC 8824-1:1998/Amd.1:1999, *Information technology – Abstract Syntax Notation One (ASN.1): Specification of basic notation – Amendment 1: Relative object identifiers.*

[6] ITU-T Recommendation X.691 (1997) | ISO/IEC 8825-2:1998, *Information technology – ASN.1 encoding rules: Specification of Packed Encoding Rules (PER).*

G.5 System Models

Annex G does not mandate a specific system architecture among administrative domains or within an administrative domain. The following subclauses will provide some sample architectures, but these are to be viewed as illustrative rather than exhaustive.

In general, an administrative domain is viewed as consisting of any number of zones and any number of border elements. Remember that a border element is a functional element that may exist together with any other H.323 element. Figure G.2 shows some examples of border element implementations in combination with other elements.

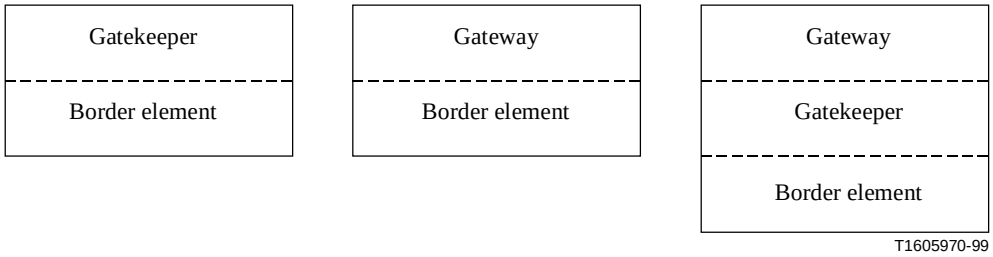

Figure G.2/H.225.0 – Border Element Placement Examples

The relationship among administrative domains may be any of a variety of organizations. The following subclauses indicate example relationships.

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.

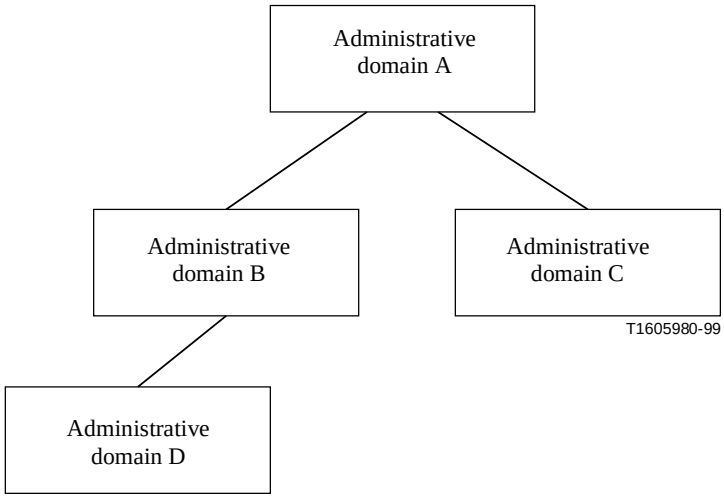

Figure G.3/H.225.0 – Sample Hierarchical Organization

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.

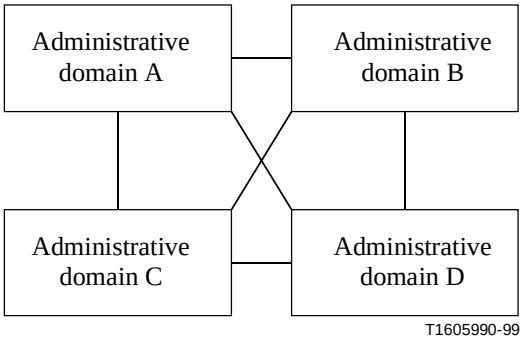

Figure G.4/H.225.0 – Sample Distributed Organization

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.

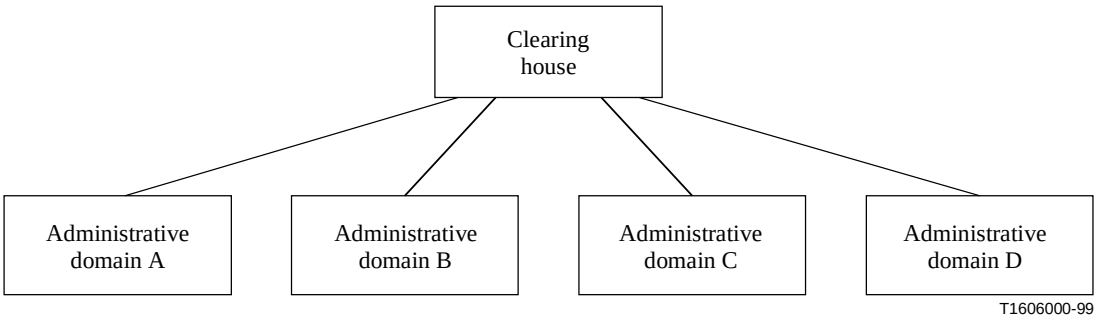

Figure G.5/H.225.0 – Sample Clearing House Organization

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.

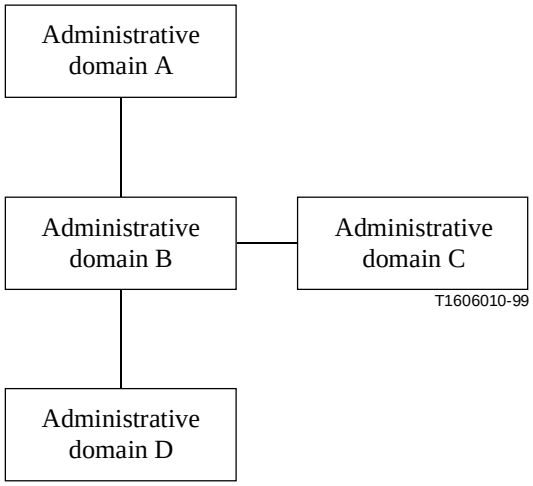

Figure G.6/H.225.0 – Aggregation Point Example

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 origination administrative domain. The algorithm employed to select the destination administrative domain is an implementation matter.

G.6 Addressing Conventions

In order to provide interoperability between domains, it is important that the addressing formats sent in H.323 messages are understood by the receiving system. A border element shall support both the email-id and partyNumber (using PublicNumber with PublicTypeOfNumber of internationalNumber) types of AliasAddress. Note that this requirement implies support of H.225.0 (1998) or later. When communicating with other border elements, only the email-id and partyNumber types of AliasAddress should be used in the destinationAddress field of an LRQ or Setup message unless there has been prior agreement between the administrative domains concerned. For example, if a group of administrative domains have agreed on the interpretation of private local numbers then these numbers may be used in messages between them.

G.7 Operation

G.7.1 Address Templates and Descriptors

An address template ("template" for short) defines a set of AliasAddress identifiers, pricing information to complete calls to those addresses, and the protocol to be used in reaching addresses in that set. An administrative domain advertises templates to indicate the calls it can resolve. Templates are grouped together by an identifier known as a "descriptor". Once a template is grouped by a descriptor, any change to a template under that descriptor implies a change to the descriptor "group". Template information may allow the aggregation of addressing information if the addressing scheme is arranged in some hierarchical or routable manner (for example, a given zone might handle 1303538*, meaning all telephone numbers that begin with 1303538). (Note that since "*" is a meaningful character, the template actually includes a Boolean flag to indicate whether the address is specific or not. These examples use "*" to indicate a wild card, but the actual representation in the template is through the Boolean flag.)

Template examples include:

"For 1 555 123 4567	send AccessRequest message to border element A".
"For 1 555 987*	send AccessRequest message to border element B".
"For 1 555 987 6543	send Setup message to gateway X".
"For*@example.org	send AccessRequest message to border element A".
"For 1*	send AccessRequest message to border element B".
"For private 31*	send AccessRequest message to border element C".
"For 44 171 112*"	doesn't exist".

A border element obtains templates in these ways:

- static configuration;
- receiving descriptors from other border elements in response to general requests;
- receiving responses to specific queries.

G.7.1.1 Static Configuration

A border element will maintain templates for all the zones for which it is responsible. These templates may be explicitly provisioned in the border element, or these templates may be formed by summarizing information obtained from gatekeepers within its domain. The border element may make this information available to other border elements via responses to requests. An administrative domain may choose the level of detail to be provided by its border elements. Examples include:

- A border element that wishes to hide internal structure might provide one descriptor (with an indication to send an AccessRequest message) which describes its whole zone and refers to a gatekeeper which will handle all incoming calls.
- A border element which 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 which is on a fire wall (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 "Send AccessRequest" which indicate the border element which should be used to contact the other administrative domain.
- A clearing house border element (such as one which has a complete copy of 44) might hold a template marked "Send Access Request" for each administrative domain within 44.

Border elements need not keep a copy of the whole database. If a border element does not hold a copy of the whole database, then it should contain statically configured "Send AccessRequest" templates indicating a clearing house border element which will be used to resolve other queries.

G.7.1.2 Receiving Descriptors

A border element may request the statically configured templates from another border element. The response to the request is decided by the border element from which the templates are being requested.

To request a transfer, the border element sends a DescriptorRequest message specifying the descriptors it wishes to receive. If the owning border element is able to transfer the descriptors, it responds with a DescriptorConfirmation message specifying all the templates.

The requesting border element may cache a copy of a template received in this manner until the template's lifetime expires, at which point the border element should delete its copy of the template. If the owning border element changes its statically configured templates before their lifetime has expired, then it shall send a DescriptorUpdate message to those border elements of which it is aware. A border 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 border element (a border 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 border 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 border element receives the AccessRequest message without per-call information and policy is to require per-call information, the border element shall reply with an AccessRejection message with a reason of **needCallInformation**.

A border element may send a DescriptorUpdate message to other known border elements, or the border element may multicast a DescriptorUpdate message. If a DescriptorUpdate message is multicast, the border element should consider the scope of the multicast. The DescriptorUpdate message can 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.7.1.3 Receiving Responses to Specific Queries

A border element may send an AccessRequest message to another border 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 border 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 border element shall return all matching templates. If the destination border element is actually responsible for the alias address specified, the border element will usually respond with a template indicating that either an AccessRequest or Setup message should be sent. If the destination border element is a clearing house, it will normally respond with a template indicating that the AccessRequest message should be sent.

The destination border element may also add templates to the response which 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 which is tightly coupled with a fire wall 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 border element may cache a template received in an AccessConfirmation until its lifetime expires.

G.7.2 Discovery of a Border Element or Set of Border Elements

G.7.2.1 Static

A border element may have an administered set of other border elements which it may contact for address resolution. This administered set may be defined through a set of bilateral agreements between the administrative domains and other administrative domains. The administrative domains may optionally utilize the service of a clearing house.

G.7.2.2 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 synthesise a "Send AccessRequest" template which can be used during the resolution process. Templates synthesised from DNS requests should not be cached for longer than the lifetime provided in the DNS response.

G.7.2.3 Other Methods

The use of other methods to locate another border element are for further study.

G.7.3 Resolution Procedures

G.7.3.1 Resolution Procedure Within Administrative Domain

When a border 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 ("Send Setup" is better than "Send AccessRequest").

If multiple templates satisfy the request, then the border element shall return all matching templates.

If the template selection procedure produces no templates marked as "Send Setup", then the border element sends an AccessRequest message with a specific destination address to the address specified in the template. When it gets an answer from the border element, it may store that in its cache and return to the requester the address to which to send the Setup message.

G.7.3.2 Resolution Procedure Between Administrative Domains

When a border element receives an AccessRequest, it searches through the templates in its cache and finds those which match the address in the query.

If more than one template matches, they are first sorted by wildcard length (more specific templates are better). They are then sorted by the message type specified ("Send Setup" is better than "Send AccessRequest"). In each case all templates other than the most specific match are discarded.

If the matched templates are marked as "Send AccessRequest" 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 border 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 a border element of a third administrative domain (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 which it has found (these will have the same address and message type fields) and any other templates which it considers will be 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.7.4 Usage Information Exchange

Administrative domains may request other domains 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 usage indications may be sent for the same call, each one with more up-to-date information.

Usage Indications may be exchanged only if the two border elements have service relationship between them.

UsageIndication requests shall be sent when a border element requires that, either in the templates for which it serves as contact, or by indicating that in either one of the UsageRequest, AccessRequest, ValidationRequest and ValidationConfirmation messages sent in the context of the call for which UsageIndication is required.

G.8 Protocol

Messages in the Annex G protocol may be sent over an unreliable transport service (e.g. UDP) or a reliable transport service (e.g. TCP) to a well-known address. On IP networks, the well-known port 2099 should be used for both TCP and UDP, unless another port has been communicated to the sender. Border elements shall listen on both TCP and UDP ports.

When messages are sent over the reliable transport service, multiple messages may be sent within the boundaries defined by the reliable transport protocol data unit (PDU) as long as whole messages are sent. (In IP implementations, as outlined in Appendix IV/H.225.0, this PDU is defined by TPKT.)

When using unreliable transport service, request messages may be retransmitted. The default value of the retransmission timer should determined by an adaptive delay sensitive method (such as the one used by the TCP protocol). Exponential backoff shall be used for subsequent retransmissions. The number of retransmissions shall not exceed 5. Responses shall not be retransmitted.

In UDP IP implementations, messages shall also be prefixed with TPKT headers, to enable multiple messages per packet. The UDP packet length field shall hold the total length of the payload, including all the messages and their TPKT headers.

G.8.1 Security Considerations

When authentication, integrity, and encryption is desired for messages exchanged between border elements, the operation of IP security shall be followed as described in IETF RFC 1825 ("Security Architecture for the Internet Protocol"), including either, or both, of IETF RFC 1826 ("IP Authentication Header"), and IETF RFC 1827 ("IP Encapsulating Security Payload (ESP)").

Where appropriate, the procedures and constructs of H.235 shall be utilized to support application-level security. Specifically, the token formats and authentication exchanges shall be used. Tokens and crypto-tokens received in response messages should be used in a subsequent related request.

G.8.2 Message Definitions

Each message contains a set of common fields in addition to the message-specific information. The common fields are:

Field	Description
sequenceNumber	Each request or update message contains a unique sequence number. The message sent in response to a request message (a confirmation or rejection message) uses the sequence number from the request message. Retransmitted messages shall have the same sequence number.
ReplyAddress	This is the address to which to send the reply to a request message. Any request message shall include a replyAddress, unless the request was sent over a bidirectional connection-oriented transport (e.g. TCP). Any message other than a request message shall not include a replyAddress.
Version	Protocol version in use by the sender of this message.
HopCount	This defines the number of border elements through which this message may propagate. When a border element receives this message and decides that the message should be forwarded on to another border element, it first decrements *hopCount*. If *hopCount* is then greater than 0, the border element inserts the new hop count value into the message to be forwarded. If *hopCount* has reached 0, the border element shall not forward the message. If the message is a request, the border element should respond with a confirmation message with any applicable information. If no information is available, the border element should respond with a rejection message.
IntegrityCheckValue	Provides improved message integrity/message authentication. 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 each byte of this field shall be set to zero. After computation, the sender puts the computed integrity check value in the integrityCheckValue field and transmits the 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.
NonStandard	Non-standard information.

G.8.2.1 Descriptor

The Descriptor is not a message, but is rather a message element used to label a set of templates.

The Descriptor contains the following information:

Field	Description
DescriptorInfo	This holds a unique identifier for the descriptor and the time it was last changed (see Descriptor Information below).
Templates	This is a set of templates which define the addresses this descriptor can resolve.
GatekeeperID	This is a text identifier that indicates the owner of the descriptor (i.e. the gatekeeper that created this message).

G.8.2.2 Descriptor Information

Descriptor information uniquely identifies the descriptor and indicates the last time the descriptor changed.

Field	Description
DescriptorID	This is a globally unique identifier used to identify this descriptor from among many possible descriptors.
LastChanged	This is the date and time this descriptor was last changed.

G.8.2.3 Address Template

The Address Template describes a set of one or more alias addresses. The Template is not a message, but is an element used as a building block for other elements. The Template consists of other structures, which are described in the following subclauses.

Field	Description
Pattern	This is a list of patterns (see Pattern below).
RouteInfo	This is a list of route information for this template (see Route Information below).
TimeToLive	This indicates the time, expressed in seconds, for which this template is valid.

G.8.2.3.1 Route Information

The route information structure found in the *template* (the *routeInfo* field) contains the following:

Field	Description
MessageType	This indicates the type of message to send when attempting to resolve a specific address within this template. Possibilities are sendAccessRequest, sendSetup, or nonExistent (indicates that the address does not exist).
CallSpecific	If set to TRUE, authorization is requested for each call to this route, implying that the AccessRequest message shall include the call information. This boolean field has meaning only when *messageType* is sendAccessRequest; otherwise, *callSpecific* shall be set to FALSE.

UsageSpec	If present, this specifies the UsageIndication messages that shall be sent regarding the calls to this route.
PriceInfo	This is a list of pricing information for this particular route (see Pricing Information below). Note that multiple gateways with different pricing structures would be described in multiple *RouteInformation* structures.
Contacts	This is contact information for the element that will accept the message as specified in the *messageType* field of routeInfo. The contact information may be provided as a list of possible contacts (see Contact Information description below).
Type	This indicates the type of endpoint that can serve the call. For gatekeeper routed cases, this indicates the types of endpoints served by the gatekeeper rather than the gatekeeper itself.

G.8.2.3.2 Pricing Information

Pricing information appears as an element in the Route Information structure (the priceInfo field). Pricing information is defined through the PriceInfoSpec and PriceElement structures.

The PriceInfoSpec structure contains the following fields:

Field	Description
Currency	This is an ISO 4217 currency designator.
CurrencyScale	This is the number of places to shift the implied radix point to the left. For example, when *currency* is specified as USD, a *currencyScale* of 2 would indicate that the amount in *priceElement* is expressed in United States cents.
ValidFrom	This is the date and time from which this information is valid.
ValidUntil	This is the date and time at which this information expires.
HoursFrom	This is the time of day when this rate starts.
HoursUntil	This is the time of day when this rate ends. It may be less than *hoursFrom*, indicating a rate which spans 0000.
PriceElement	This is an optional list of PriceElements which sum to effect the pricing.
PriceFormula	This is an optional string containing a pricing formula used as an alternative to the structured PriceElement.

The PriceElement structure contains the following fields:

Field	Description
Amount	This is the meter increment. The meter increments once for each *quantum* or fraction of *quantum*.
Quantum	This is the number of units for which *amount* applies. For example, a value of 60, with *units* in seconds, indicates that the call is priced per minute or fraction of minute. If the *units* field is set to either of *initial*, *minimum* or *maximum* values**,** then the *quantum* field is irrelevant, and its value shall be ignored by the recipient.

Units	This is the type of unit in which quantum is expressed: • Seconds – Seconds of call duration. • packets – Packets transmitted or received. • bytes – Bytes transmitted or received. • initial – An initial connect charge. • minimum – A minimum call charge. • maximum – A maximum call charge.

G.8.2.3.3 Contact Information

The Contact Information structure is an element of the Route Information structure (the *contacts* field).

Field	Description
transportAddress	This is the address (e.g. transport address or URL) to which to send the message specified in the *messageType* field of the Route Information structure. Whenever possible, a transport address shall be used.
Priority	When multiple contacts are listed, the *priority* field specifies the order in which the multiple contacts should be tried. Contacts in the list can share a priority, for example if there is no preference on the order in which the contacts should be tried. A priority of 0 indicates the highest priority (first choice).
TransportQoS	Indicates where the responsibility lies for resource reservation for the call made through this contact.
Security	Security mechanism in describing order of preference to be used when communicating with contact.
AccessTokens	This is a set of tokens that shall be passed in the message to this contact (Setup or AccessRequest). These tokens shall also be sent in subsequent UsageIndication messages pertaining to the calls using this template.

G.8.2.3.4 Pattern

The Pattern structure appears in the Address Template. The Pattern allows specification of an alias address, a wildcarded alias address, or a range of alias addresses:

Field	Description
Specific	This is a specific alias address.
Wildcard	This some hierarchical definition that represents possible expansion of the string. For 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.
Range	This is a range of addresses, including the indicated start and end of range.

G.8.2.4 Common Structures

The structures defined in this subclause appear in many of the messages.

G.8.2.4.1 AlternateBE

Field	Description
ContactAddress	This is the alternate border element's transport address (the address to which to send Annex G messages).
Priority	When multiple alternates are listed, the *priority* field specifies the order in which the multiple alternates should be tried. Alternates in the list can share a priority, for example if there is no preference on the order in which the alternates should be tried. A priority of 0 indicates the highest priority (first choice).
ElementIdentifier	This alternate border element uses this unicode string as an identifier.

G.8.2.4.2 PartyInformation

This structure contains information about a party of the call (either source or destination).

Field	Description
LogicalAddress	E-mail or E.164 formatted addresses that identify the party.
DomainIdentifier	An alias address identifying the AD which originated, or terminated the call. In case where multiple domains are involved in placing a call, then the domain that served as the call origination or termination from the sender's perspective should be stated.
TransportAddress	This is the transport address of the endpoint.
EndpointType	This indicates details about the endpoint type and capabilities.
UserInfo	This is information regarding the user behind the call. This may include identification in e-mail or PIN number format, and possible authentication credentials.
TimeZone	This is the Time zone of the party, as relevant for pricing purposes. If the originating party is a gateway, then the time zone of the gateway has to be conveyed. Described in seconds relative to UTC.

G.8.2.4.3 CallInformation

Information for identifying a specific call.

Field	Description
CallIdentifier	This provides unique identification of the call. This shall be the callIdentifier associated with the same call as in RAS and call signalling messages.
ConferenceID	This provides unique identification of the conference to which the call belongs. This shall be the conferenceID associated with the same call as in RAS and call signalling messages.

G.8.2.4.4 UserInformation

Information for identifying the user on any party of the call.

Field	Description
UserIdentifier	Uniquely identifies the user.
UserAuthenticator	Encrypted tokens for secure authentication.

G.8.2.4.5 Usage Specification

This element describes the required parameters needed to be reported in the UsageIndication messages. The calls for which this specification applies is determined by the context of the message containing the *UsageSpecification* element.

Field	Description
SendTo	Border element to send the UsageIndication messages to. Since the sender should have service relationship with that border element, this is the element identifier returned in the ServiceConfirmation message.
When	Specifies the stages of the call, and the frequency, at which the indications should be sent: • Never – Stop sending messages. • Start – When the call begins. • End – By the end of the call, or thereafter. • Period – Periodically, during the call lifetime. The period is measured in seconds. • Failure – Report failed call attempts.
Required	A list of identifiers for fields that *must* be present in the *UsageIndication* messages. The sender of the usage information shall reject or ignore the message containing this message, if it cannot supply these fields.
Preferred	A list of identifiers for fields that *should* be present in the *UsageIndication* messages.

G.8.2.4.6 Security Mode

This element describes a specific security profile to be used for Annex G communication.

Field	Description
Authentication	This indicates the authentication mechanism to be used. The authentication mechanism must be chosen from the set provided in the ServiceRequest message.
Integrity	This indicates the integrity mechanism to be used. If present, all subsequent messages shall populate the *integrityCheckValue* field, in this case, the *AuthenticationMode* describes the way the secret keys are generated (DH exchange, or *a priori*).
AlgorithmOID	This indicates the encryption algorithm for the security mechanism.

G.8.2.5 Service Request

A border element may send a ServiceRequest message to another border element to establish a service relationship. The relationship defines the security mechanisms to be used between the border elements and allows identification of alternate, or backup, border elements. Note that the relationship is a one-way relationship. The security negotiated between the 2 border elements is used for requests sent by the border element that sent the ServiceRequest and for responses sent by the recipient of the ServiceRequest. Session keys may be generated during the process of service relationship establishment. The keys will be valid through the lifetime of the service relationship. Tokens may be used for that purpose, as defined in Recommendation H.235.

The recipient of the ServiceRequest may indicate alternate border elements that the sender of ServiceRequest may try for backup service. Establishment of a service relationship is mandatory for UsageIndication message exchanges. Otherwise, it is an optional procedure, although a border element's policy may require such a relationship.

A border element may send a ServiceRequest message to a border element with which it has an existing relationship, with the intent that the terms of the original relationship be terminated and replaced with the new terms. Service relationships may have limited time to live. A border element may refresh the relationship by sending a new Service Request.

Field	Description
ElementIdentifier	A string that identifies the BE that sends the request.
DomainIdentifier	The AD that requests the service relationship.
SecutrityCapability	Set of security mechanisms that this border element can support.
TimeToLive	The suggested lifetime in seconds for the service relationship. If not present, infinite lifetime is assumed.

G.8.2.6 Service Confirmation

A border element in receipt of a ServiceRequest message responds with a ServiceConfirmation message to indicate that it agrees to establish a service relationship. If the border element already has a service relationship with the border element that sent the ServiceRequest message, sending ServiceConfirmation indicates that the terms of the original relationship are terminated and replaced with the new terms.

Field	Description
ElementIdentifier	This is a string that identifies the border element.
Alternates	This is a list of alternate border elements that may be contacted in the event that this border element fails to respond.
DomainIdentifier	The AD that responds to the request.
SecurityMode	This indicates the security mechanism to be used for this service relationship. The security mechanism must be chosen from the set provided in the ServiceRequest message.
TimeToLive	The lifetime in seconds of the service relationship as determined by the serving border element.

G.8.2.7 Service Rejection

A border element in receipt of a ServiceRequest message responds with a ServiceRejection message to indicate that it declines to establish a service relationship. If the border element already has a service relationship with the border element that sent the ServiceRequest message, sending ServiceRejection indicates that the proposed new terms have been rejected, but the terms of the original relationship remain.

Field	Description
Reason	This is the reason the border element rejected the ServiceRequest. Choices are: • ServiceUnavailable – This border element is not currently available for service. • ServiceRedirected – The list of alternate border elements should be attempted. • Security – This border element cannot support any of the security mechanisms proposed in the ServiceRequest message. • Continue – Indicates the subsequent ServiceRequest message be sent, in order to continue multiple stage key exchange process. • Undefined – The reason for rejecting the ServiceRequest does not match any of the other choices.
Alternates	This is a list of alternate border elements that might be able to honour the ServiceRequest. If the *reason* is *serviceRedirected*, at least one alternate should be provided.

G.8.2.8 Service Release

Either border element in a service relationship may terminate the relationship by sending the ServiceRelease message.

Field	Description
Reason	This is the reason this border element terminated the service relationship. Choices are: • OutOfService – The border element is going out of service. • Maintenance – The border element is being taken out of service for maintenance. • Terminated – The border element has decided to terminate the relationship. • Expired – The time-to-live for the service relationship has elapsed.
Alternates	This is a list of alternate border elements that might be able to establish a service relationship.

G.8.2.9 Descriptor Request

The DescriptorRequest message allows an entity to query a border element for specific descriptors.

Field	Description
DescriptorID	This identifies one or more particular descriptors requested by the sender of this message.

G.8.2.10 Descriptor Confirmation

The DescriptorConfirmation message is a border element's positive response to a DescriptorRequest, when the border element can interpret the request and implementation rules allow information exchange.

Field	Description
Descriptors	This is the *descriptors* described above.

G.8.2.11 Descriptor Rejection

A border element can reject a descriptor request for a variety of reasons.

Field	Description
Reason	This is the reason the DescriptorRequest was rejected. Choices are: • PacketSizeExceeded – The reply would exceed the maximum packet size, so the requester should send the request using a different transport mechanism (e.g. use TCP instead of UDP). • illegalID – The recipient of the DescriptorRequest has no record of the requested descriptor. • security – The DescriptorRequest did not meet the recipient's security requirements. • HopCountExceeded – The hop count reached zero and no information is available. • unavailable – The recipient cannot provide descriptors. Static or out-of-band provisioning method should be used. • noServiceRelationship – The recipient will exchange this information only after establishment of a service relationship. • undefined – The reason for rejecting the DescriptorRequest does not match the other choices.
DescriptorID	This identifies the specific descriptor for this response.

G.8.2.12 Descriptor ID Request

The DescriptorIDRequest allows an entity to query a border element for the list of descriptor identifiers within the border element's administrative domain.

G.8.2.13 Descriptor ID Confirmation

A DescriptorIDConfirmation message is a border element's positive response to the DescriptorIDRequest message. A border element in receipt of a DescriptorIDConfirmation message may send the DescriptorRequest message to request transmission of the descriptors.

Field	Description
DescriptorInfo	This is a list of descriptor information, where each entry in the list uniquely identifies the descriptor and the time it last changed.

G.8.2.14 Descriptor ID Rejection

A border element can reject a DescriptorIDRequest for a variety of reasons.

Field	Description
Reason	This indicates the reason for rejecting the request. Choices are: • noDescriptors – This indicates that the border element has no descriptors to report. • security – The DescriptorIDRequest did not meet the recipient's security requirements. • hopCountExceeded – The hop count reached zero and no information is available. • unavailable – The recipient cannot provide descriptors. Static or out-of-band provisioning method should be used. • NoServiceRelationship – The recipient will exchange this information only after establishment of a service relationship. • undefined – The reason for rejecting the DescriptorIDRequest does not match the other choices.

G.8.2.15 Descriptor Update

The DescriptorUpdate message is a border element's notification that address information has changed. A border element may also send the DescriptorUpdate message during initialization. A border element in receipt of the DescriptorUpdate may request information from the element identified in the DescriptorUpdate.

Field	Description
Sender	An element in receipt of the DescriptorUpdate may send a request to this address (e.g. transport address or URL).
UpdateInfo	This is a list of updates. Each entry in the list provides either the descriptor or the descriptor identifier that was updated. Each entry in the list also indicates whether the descriptor was changed, added or deleted.

G.8.2.16 Descriptor Update Acknowledgement

A border element should acknowledge receipt of a DescriptorUpdate message by sending the DescriptorUpdateAck message. The sqeuence number used in the acknowledgement should be the same as the sequence number received in the DescriptorUpdate message. A border element should not acknowledge a DescriptorUpdate message that arrives over multicast.

G.8.2.17 Access Request

A border element can send an AccessRequest message to another border element to ask for resolution of a specific alias address.

Field	Description
DestinationInfo	This is the address to be resolved.
SourceInfo	This is information about the originating party of the call to which access is requested.

CallInfo	This provides identification of the particular call for which access authorization is requested. If not present, then the request is for indefinite calls to the specified destinations.
UsageSpec	This indicates the usage messages that the originating party requests the answering party to send regarding the call requested in this message. Applies only if *CallInfo* is present.

G.8.2.18 Access Confirmation

A border element returns in the AccessConfirmation message the information requested in the AccessRequest message.

Field	Description
Templates	This is a list of tempates which match the attributes of the AccessRequest.
PartialResponse	If TRUE, this message contains some fraction of the available information. The entire information was not sent because it would exceed the packet size. The entire information should be retrieved using another transport type (e.g. TCP).

G.8.2.19 Access Rejection

A border element can reject an AccessRequest for a variety of reasons.

Field	Description
Reason	This is the reason for rejecting the request. Choices are: • NoMatch – The destination specified in the AccessRequest cannot be resolved. • PacketSizeExceeded – The reply would exceed the maximum packet size, so the requester should send the request using a different transport mechanism (e.g. use TCP instead of UDP). • security – The AccessRequest did not meet the recipient's security requirements. • HopCountExceeded – The hop count reached zero and no information is available. • NoServiceRelationship – The recipient will exchange this information only after establishment of a service relationship. • CallInfoNeeded – Specific call information was not present in the request. • Undefined – The reason for rejecting the AccessRequest does not match the other choices.

G.8.2.20 Request in Process

A border element may return the RequestInProgress message to indicate that the time required by the border element to respond to a request may exceed normal expected response intervals. The sequence number shall be the same sequence number found in the request for which this message will be sent.

Field	Description
Delay	The expected length of time, expressed in milliseconds, for the border element to respond to the original request.

G.8.2.21 Non-Standard Request

The NonStandardRequest may be sent from a border element to represent a request message not defined in Annex G. The non-standard information is carried in the *nonStandard* element of *AnnexGCommonInfo*.

G.8.2.22 Non-Standard Confirmation

The NonStandardConfirmation may be sent from a border element in response to a NonStandardRequest message. The non-standard information is carried in the *nonStandard* element of *AnnexGCommonInfo*.

G.8.2.23 Non-Standard Rejection

The NonStandardRejection may be sent from a border element in response to a NonStandardRequest message. The non-standard information is carried in the *nonStandard* element of *AnnexGCommonInfo*.

Field	Description
Reason	This is the reason for rejecting the request. Choices are: • notSupported – The recipient understands that this is a NonStandardRequest, but does not understand or support the non-standard data. • noServiceRelationship – The recipient will exchange this information only after establishment of a service relationship. • undefined – The reason for rejecting the NonStandardRequest does not match the other choices.

G.8.2.24 Unknown Message Response

A border element in receipt of a message it does not understand should respond to the transmitter with the UnknownMessageResponse message. The border element should not use this message if some other Annex G message provides an appropriate response (for example, a DescriptorRejection would be the appropriate response to a DescriptorRequest with an illegal descriptor identifier).

Field	Description
unknownMessage	This is the contents of the unknown message.
Reason	This is the reason the the UnknownMessageResponse was used. Choices are: • notUnderstood – The message was not understood. • undefined – The reason for sending UnknownMessageResponse does not match any of the other choices.

G.8.2.25 Usage Request

Request the recipient to send UsageIndication messages concerning a specific call.

Field	Description
CallInfo	The call for which to send the Indication.
UsageSpec	Specifies when the indications should arrive, and what they should contain.

G.8.2.26 Usage Confirmation

The UsageConfirmation message is sent in response to a UsageRequest message to indicate that the recipient accepted the request and will send usage indications.

G.8.2.27 Usage Rejection

The UsageRejection message is sent in response to a UsageRequest message to indicate that the recipient rejected the request and will not send the usage indications subsequently.

Field	Description
Reason	This is the reason the border element rejected the UsageRequest. Choices are: • InvalidCall. • Security. • Unavailable. • noServiceRelationship. • Undefined.

G.8.2.28 Usage Indication

Report call details and usage information. This message is sent with respect to the last *UsageSpecification* element received by the BE concerning the call.

Field	Description
CallInfo	The call for which the indication applies.
AccessTokens	The access tokens for the call. These are the tokens that were received in the address template used for the call, and propagated in the AccessRequest/Setup message for the same call.
SenderRole	The role of the sender of the indication: • originator – originating party. • destination – terminating party. • nonStandard – other.
UsageCallStatus	The current status of the call: • preConnect. • callInProgress. • callEnded.
SourceAddress	E.164 or e-mail address of the caller party. In case of E.164 this designates the ANI/CLI.
DestAddress	E.164 or e-mail address for the called party.

StartTime	The time the call started in UTC format. Relevant only for calls that passed the setup stage.
EndTime	The time the call ended in UTC format. Relevant only for ended calls.
TerminationCause	The reason for the end of the call. Relevant only for ended calls.
usageInformation	Set of fields of information. Each field is represented by a *UsageField* which can be a standard or non-standard. Standard UsageFields are for future study.

G.8.2.29 Usage Indication Confirmation

The UsageIndicationConfirmation message is sent in response to a UsageIndication message, indicating that the recipient accepted the indication as reported.

G.8.2.30 Usage Indication Rejection

The UsageIndicationRejection message is sent in response to a UsageIndication message, indicating that the recipient rejected the indication and will ignore it.

Field	Description
Reason	This is the reason the border element rejected the UsageIndication message. Choices are: • InvalidCall. • Security. • NoServiceRelationship. • Undefined.

G.8.2.31 Validation Request

A border element that terminates a call can send a ValidationRequest message to another border element to verify the validity of the origination of the call.

Field	Description
DestinationInfo	Details about the destination of the call.
SourceInfo	This is information about the type of endpoint that originated the call.
CallInfo	This provides identification of the particular call for which access authorization is requested.
UsageSpec	If present, indicates the border element sending the message requests that it be sent usage indication regarding the validated call.
AccessTokens	Tokens received from the originator to prove access authorization for the call.

G.8.2.32 Validation Confirmation

Indicates that the call is validated. The requesting border element may terminate the call. The validating border element may indicate aliases to terminate the call.

Field	Description
DestinationInfo	Alternative parameters for the destination to be used by the recipient border element.
UsageSpec	If present, indicates the border element sending the confirmation requests that it be sent usage indication regarding the validated call.

G.8.2.33 Validation Rejection

Indicates the call is not valid. The requesting border element may not complete the call.

Field	Description
Reason	These are the reasons for rejecting the request. Choices are: • tokenNotValid – The access token supplied is not valid for the call. • Security – The ValidationRequest did not meet the recipient's security requirements. • HopCountExceeded – The hop count reached zero and no information is available. • MissingSourceInfo – The source information supplied was not sufficient to validate the call. • MissingDestInfo – The source inforation supplied was not sufficient to validate the call. • noServiceRelationship – The recipient will exchange this information only after establishment of a service relationship. • Undefined – The reason for rejecting the ValidationRequest does not match the other choices.

G.9 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 shown to be exchanged between a gatekeeper and a border element within the same administrative domain. This is for pure illustrative purpose, since the protocol for reference point B has not been determined (see G.1)

G.9.1 Distributed or Full Mesh

An example of a distributed network is shown in Figure G.7.

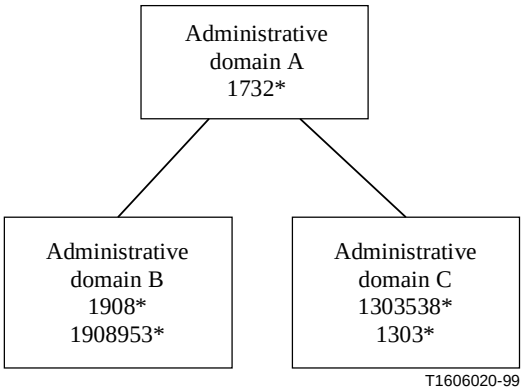

Figure G.7/H.225.0 – Distributed Network for Signalling Examples

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 "d1": Pattern = 1908* Transport address = BE_B annex g address Message type = sendAccessRequest Descriptor "d2": Pattern = 1908953* Transport address = GW_{B1} 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.
C	Descriptor "d1": Pattern = 1303538* Transport address = GK_{C1} call signal address Message type = sendSetup Descriptor "d2": Pattern = 1303* Transport address = BE_C annex g address Message type = sendAccessRequest	Calls to 1303538* will be routed through this particular gatekeeper. Calls to 1303* can be signalled directly to the destination gateway, but an AccessRequest must be sent to obtain the gateway's call signalling address.

G.9.1.1 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.

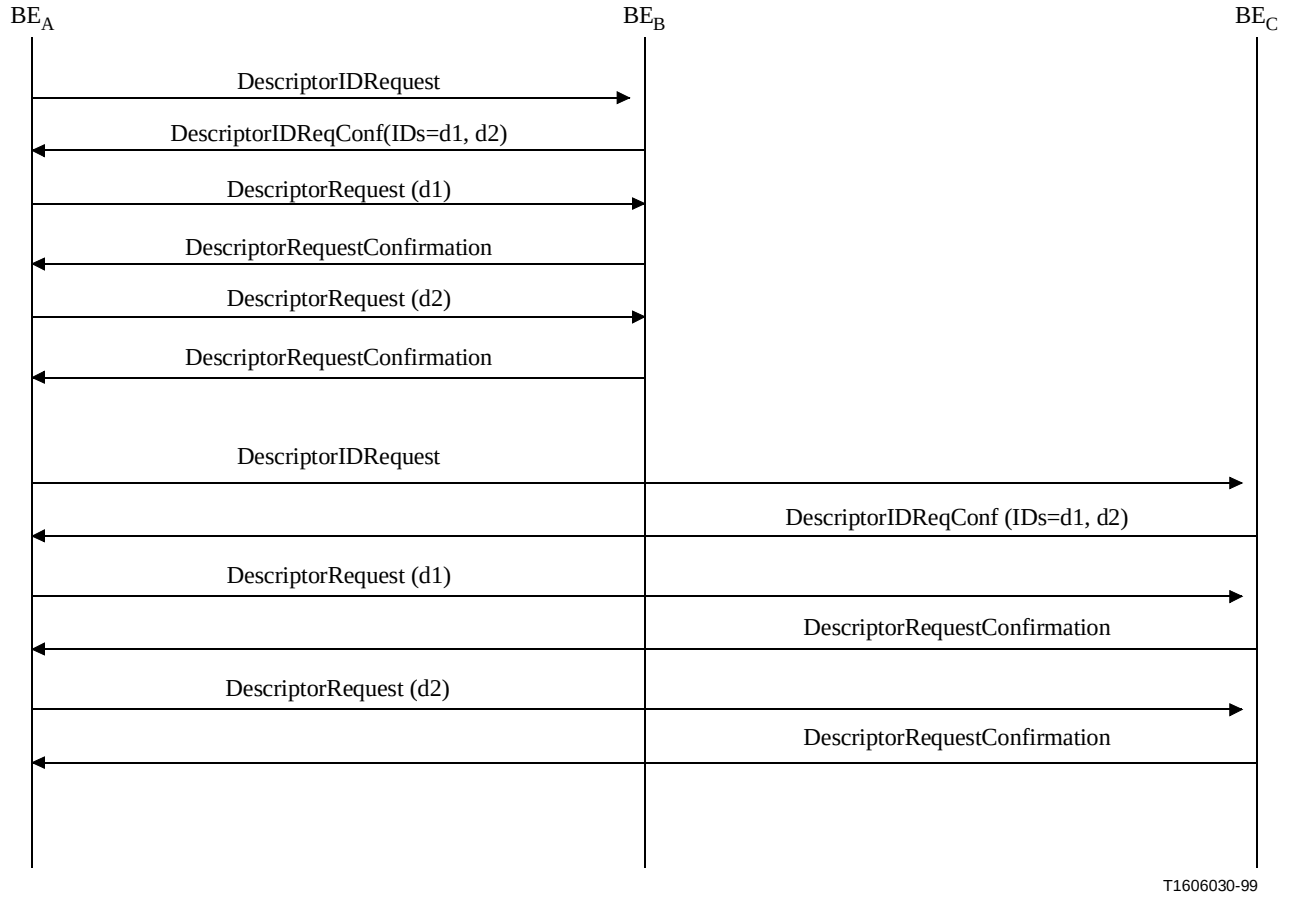

Figure G.8/H.225.0 – Example of Descriptor Exchange

Similarly, BE_B queries BE_A and BE_C, and BE_C queries BE_A and BE_B.

G.9.1.2 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 Recommendation H.323 ot its annexes.

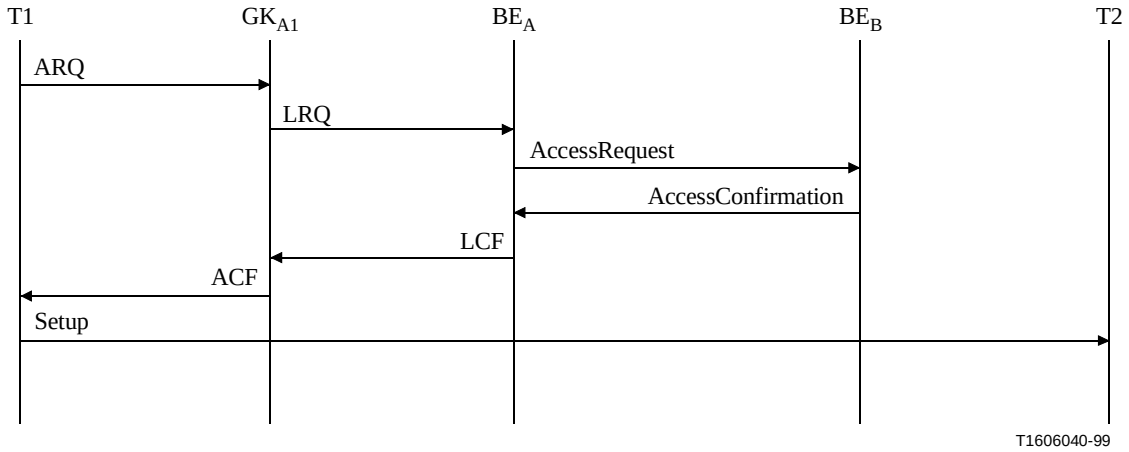

Figure G.9/H.225.0

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 which 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.

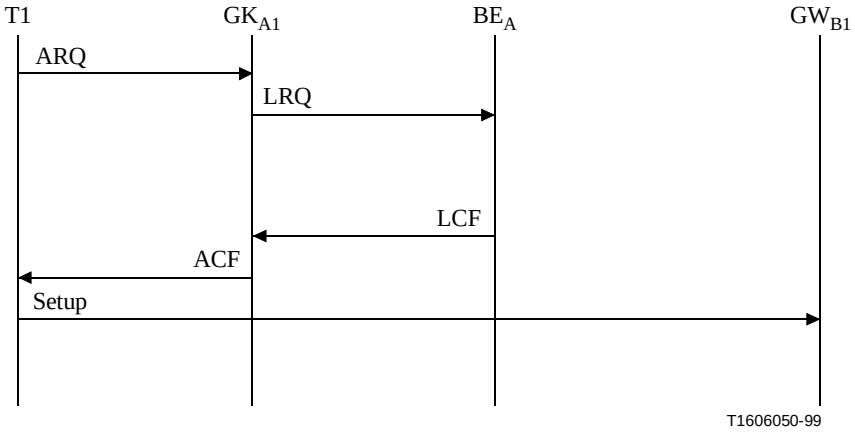

Figure G.10/H.225.0

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.

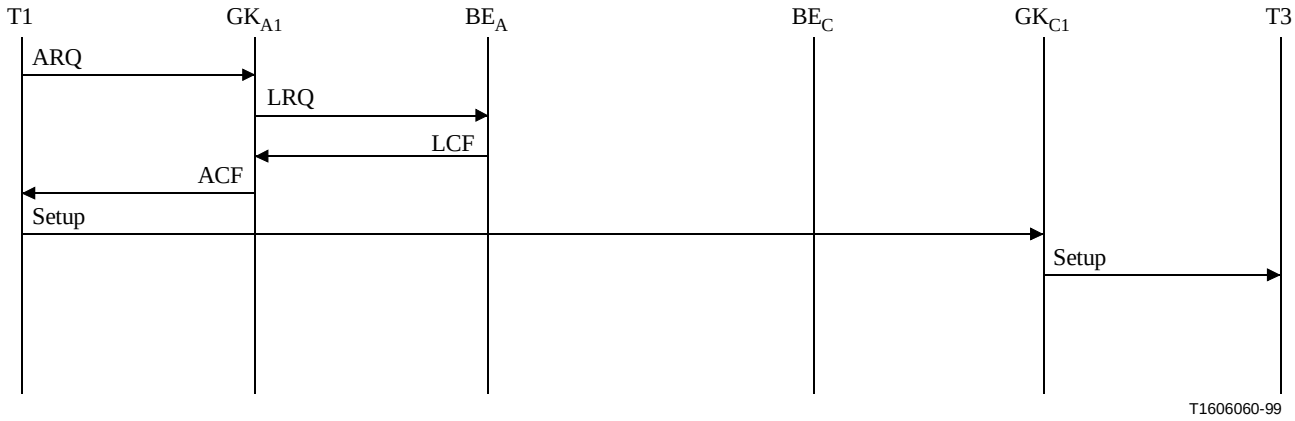

Figure G.11/H.225.0

Alternatively, T1's gatekeeper can implement the gatekeeper routed model, as shown in Figure G.12.

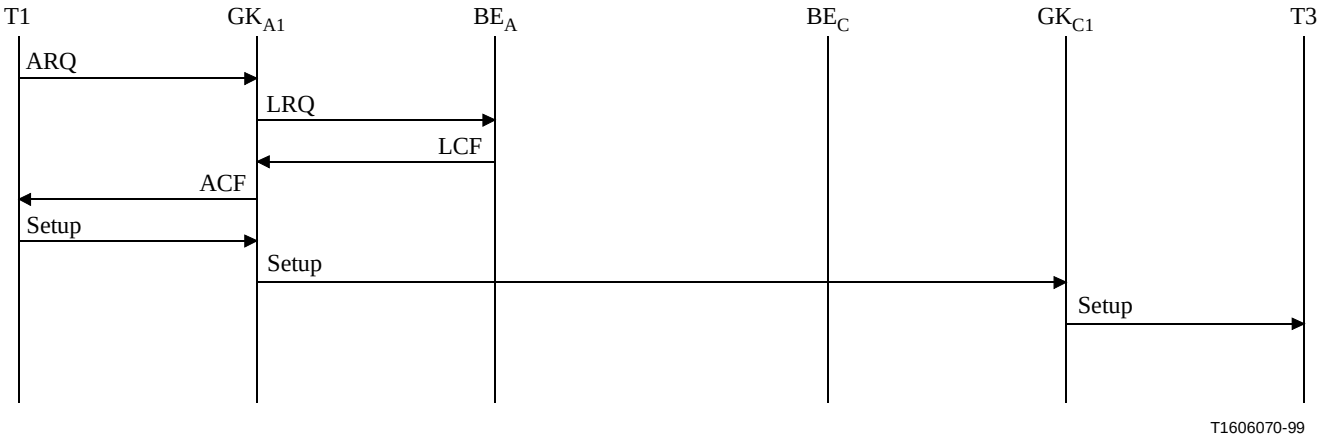

Figure G.12/H.225.0

G.9.2 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.

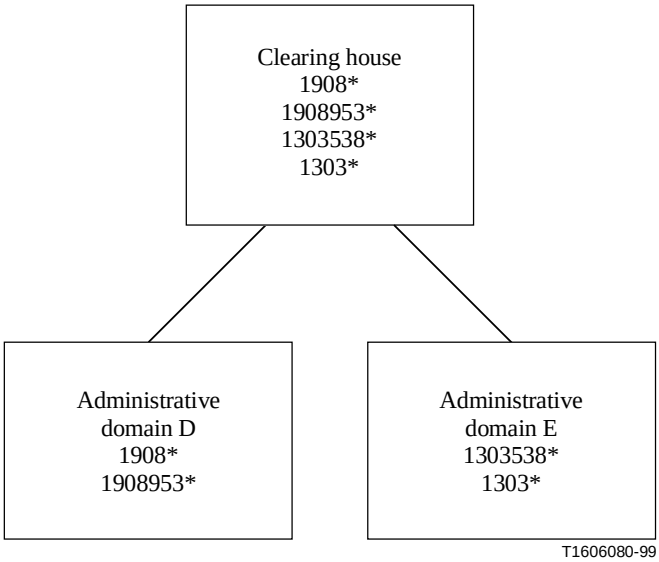

Figure G.13/H.225.0 – Sample Clearing House Configuration

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 = sendAccess Request Descriptor "d2": Pattern = 1908953* Transport address = GW_{D1} Call Signalling address Message type = sendSetup	For calls to 1908*, an Access Request 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 "d1": Pattern = 1303538* Transport address = GK_{E1} call signal address Message type = sendSetup Descriptor "d2": Pattern = 1303* Transport address = BE_E annex g address Message type = sendAccess Request	Calls to 1303538* will be routed through this particular gatekeeper. Calls to 1303* can be signalled directly to the destination gateway, but an AccessRequest must be sent to obtain the gateway's call signalling address.

CH	Descriptor "d1": Pattern = 1908* Transport address = BE_D annex g address Message type = sendAccess Request Descriptor "d2": Pattern = 1908953* Transport address = GWD_{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 = sendAccess Request	The clearing house obtains descriptors from other ADs and holds this information for distribution during descriptor exchange.

G.9.2.1 Exchange of Zone Information

In this example, a clearing house exchanges information with administrative domains which 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.

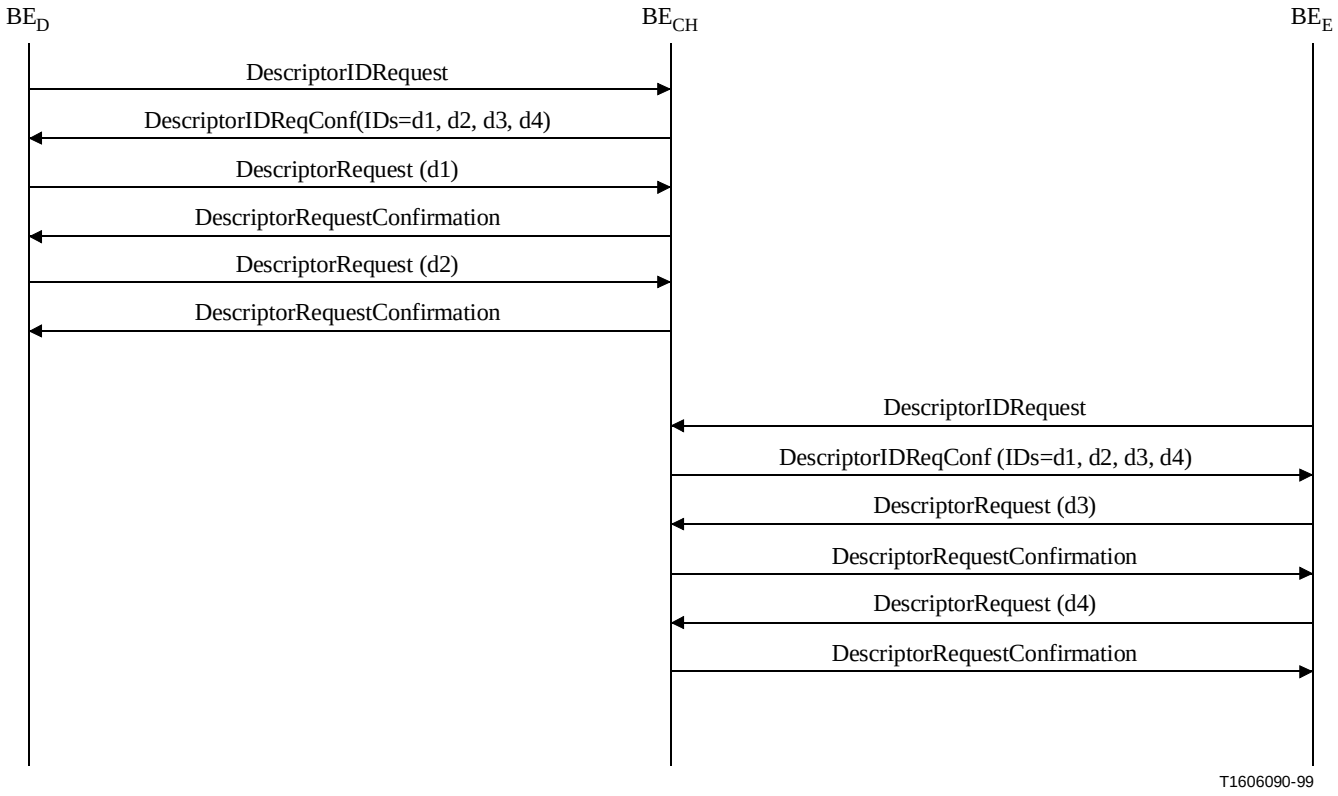

Figure G.14/H.225.0 – Example Descriptor Exchange with Clearing House

G.9.2.2 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.

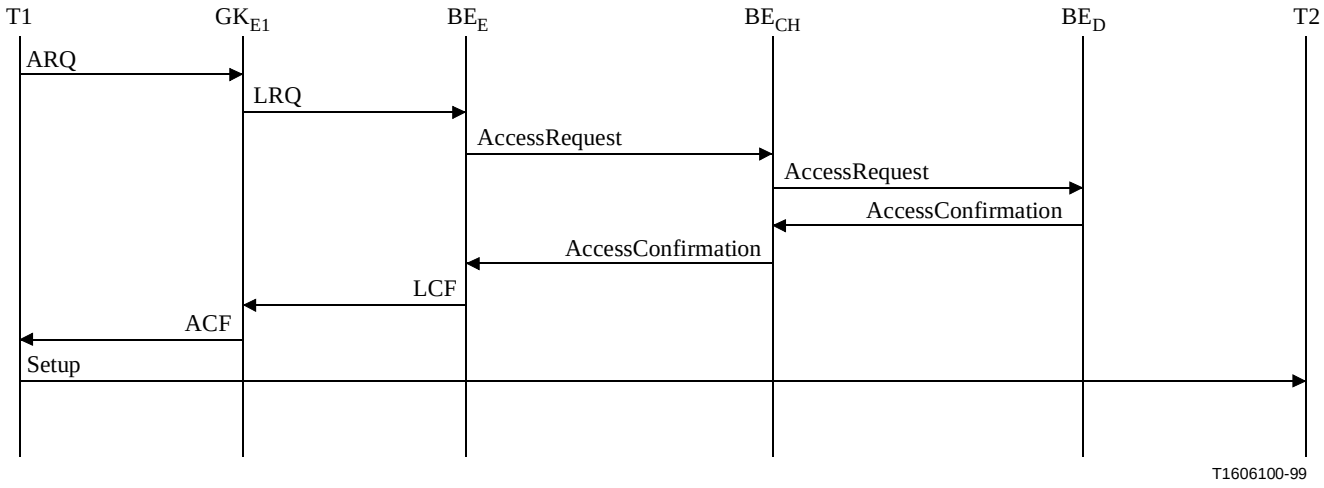

Figure G.15/H.225.0

Alternatively, T1's gatekeeper could route the call signalling, as shown in Figure G.16.

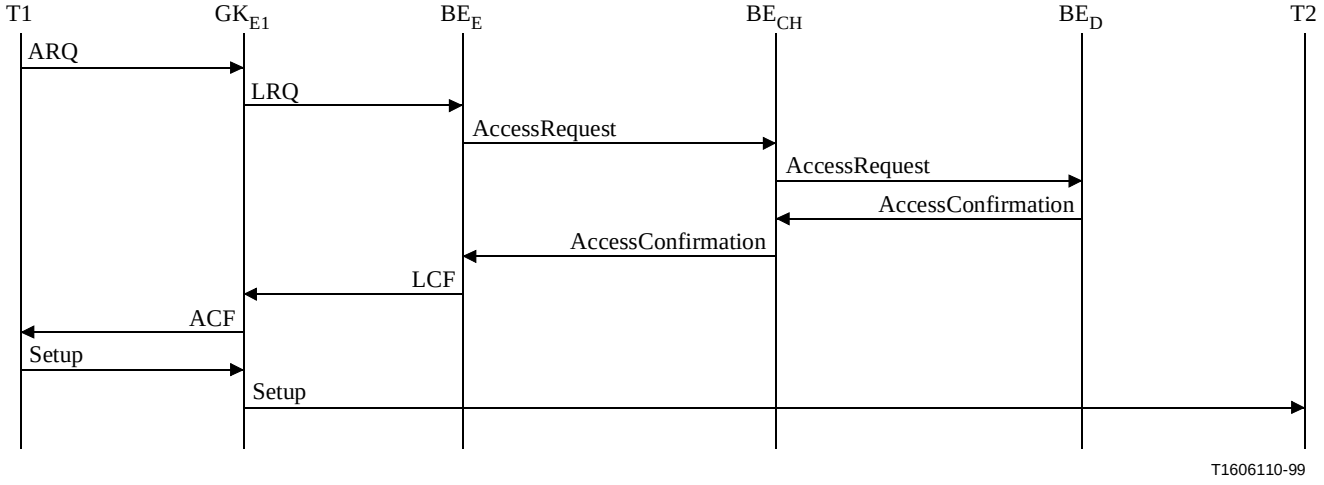

Figure G.16/H.225.0

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.

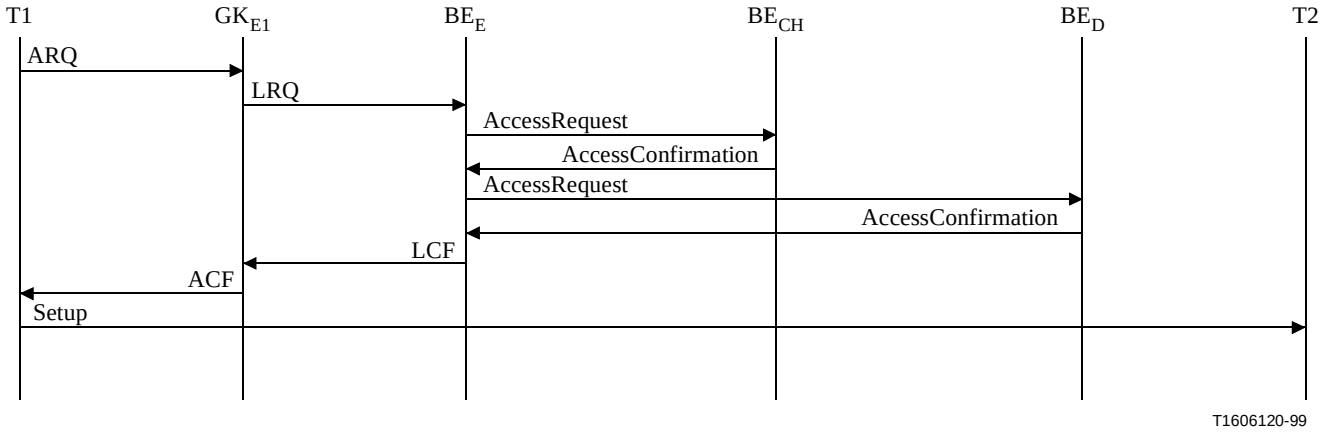

Figure G.17/H.225.0

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.

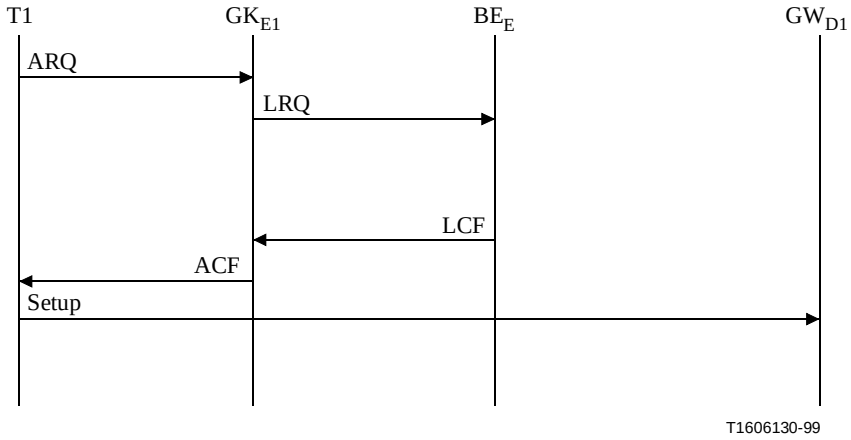

Figure G.18/H.225.0

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 Access Request 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.

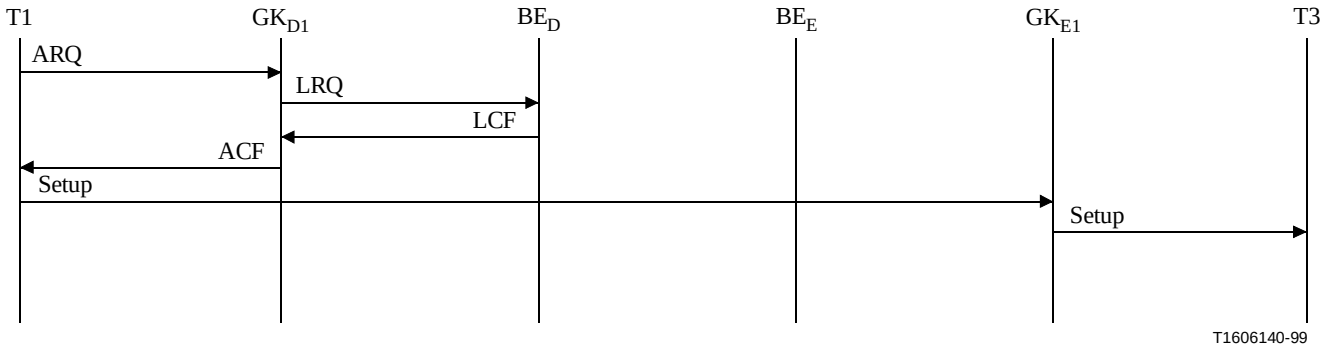

Figure G.19/H.225.0

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.

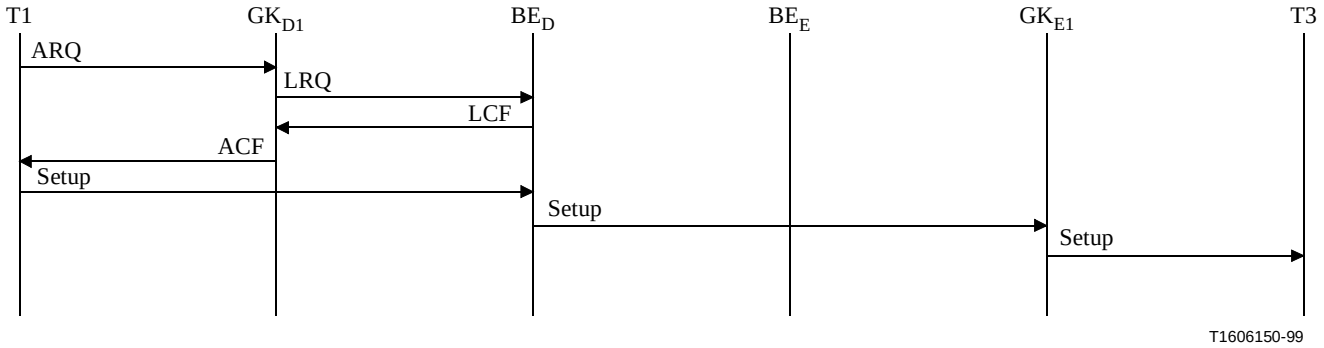

Figure G.20/H.225.0

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 usage indications for the call.

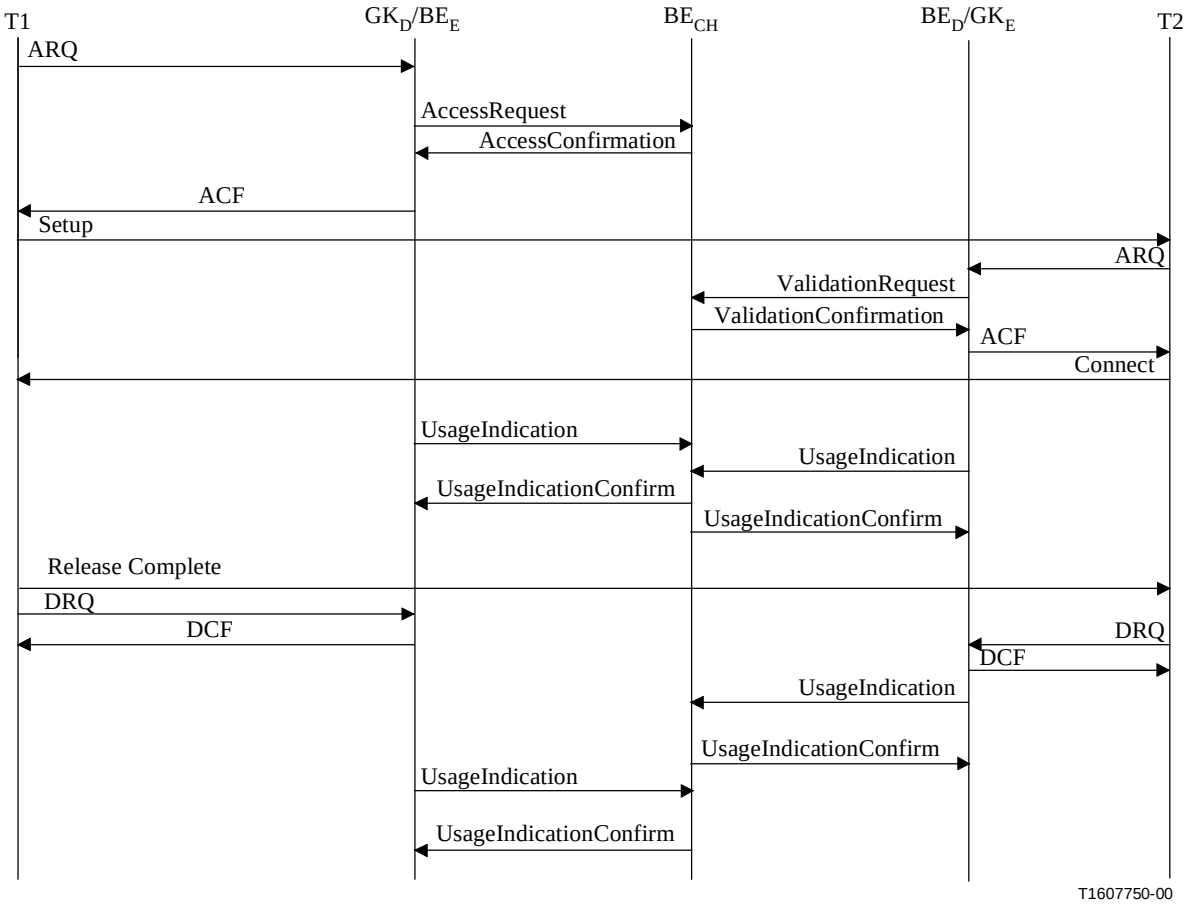

Figure G.21/H.225.0

Message Syntax

```
ANNEXG-MESSAGES DEFINITIONS AUTOMATIC TAGS ::=
BEGIN
IMPORTS
      AuthenticationMechanism,
      TimeStamp,
      ClearToken
      FROM H235-SECURITY-MESSAGES

      AliasAddress,
      TransportAddress,
      ReleaseCompleteReason,
      ConferenceIdentifier,   CallIdentifier,   CryptoH323Token,  CryptoToken,

      EndpointType,
      GatekeeperIdentifier,
      GloballyUniqueID,
      NonStandardParameter,
      NumberDigits,
      PartyNumber,
      TransportQOS,
      VendorIdentifier,
```

```
      IntegrityMechanism,
      ICV
      FROM H323-MESSAGES;

Message ::= SEQUENCE
{
      body  AnnexGMessageBody,
      common      AnnexGCommonInfo,
      ...
}

AnnexGMessageBody ::= CHOICE
{
      serviceRequest               ServiceRequest,
      serviceConfirmation          ServiceConfirmation,
      serviceRejection             ServiceRejection,
      serviceRelease               ServiceRelease,
      descriptorRequest            DescriptorRequest,
      descriptorConfirmation       DescriptorConfirmation,
      descriptorRejection          DescriptorRejection,
      descriptorIDRequest          DescriptorIDRequest,
      descriptorIDConfirmation     DescriptorIDConfirmation,
      descriptorIDRejection        DescriptorIDRejection,
      descriptorUpdate             DescriptorUpdate,
      descriptorUpdateAck          DescriptorUpdateAck,
      accessRequest                AccessRequest,
      accessConfirmation           AccessConfirmation,
      accessRejection              AccessRejection,
      requestInProgress            RequestInProgress,
      nonStandardRequest           NonStandardRequest,
      nonStandardConfirmation      NonStandardConfirmation,
      nonStandardRejection         NonStandardRejection,
      unknownMessageResponse       UnknownMessageResponse,
      usageRequest                 UsageRequest,
      usageConfirmation            UsageConfirmation,
      usageIndication              UsageIndication,
      usageIndicationConfirmation  UsageIndicationConfirmation,
      usageIndicationRejection     UsageIndicationRejection,
      usageRejection               UsageRejection,
      validationRequest            ValidationRequest,
      validationConfirmation       ValidationConfirmation,
      validationRejection          ValidationRejection,
      ...
}

AnnexGCommonInfo ::= SEQUENCE
{
      sequenceNumber          INTEGER (0..65535),
      version                 AnnexGVersion,
      hopCount                INTEGER (1..255),
      replyAddress            SEQUENCE OF TransportAddress OPTIONAL,
                              -- Must be present in request
      integrityCheckValue     ICV OPTIONAL,
      tokens                  SEQUENCE OF ClearToken OPTIONAL,
      cryptoTokens            SEQUENCE OF CryptoH323Token OPTIONAL,
      nonStandard             SEQUENCE OF NonStandardParameter OPTIONAL,
      ...
}



--
-- Annex G messages
--
```

```
ServiceRequest ::= SEQUENCE
{
      elementIdentifier       ElementIdentifier OPTIONAL,
      domainIdentifier  AliasAddress OPTIONAL,
      securityMode      SEQUENCE OF SecurityMode OPTIONAL,
      timeToLive        INTEGER (1..4294967295) OPTIONAL,
      ...
}

SecurityMode ::= SEQUENCE
{
      authentication    AuthenticationMechanism OPTIONAL,
      integrity         IntegrityMechanism OPTIONAL,
      algorithmOIDs     SEQUENCE OF OBJECT IDENTIFIER OPTIONAL,
      ...
}

ServiceConfirmation ::= SEQUENCE
{
      elementIdentifier ElementIdentifier,
      domainIdentifier  AliasAddress,
      alternates        AlternateBEInfo OPTIONAL,
      securityMode      SecurityMode OPTIONAL,
      timeToLive        INTEGER (1..4294967295) OPTIONAL,
      ...
}

ServiceRejection ::= SEQUENCE
{
      reason            ServiceRejectionReason,
      alternates        AlternateBEInfo OPTIONAL,
      ...
}

ServiceRejectionReason ::= CHOICE
{
      serviceUnavailable      NULL,
      serviceRedirected       NULL,
      security                NULL,
      continue                NULL,
      undefined               NULL,
      ...
}


ServiceRelease ::= SEQUENCE
{
      reason            ServiceReleaseReason,
      alternates        AlternateBEInfo OPTIONAL,
      ...
}

ServiceReleaseReason ::= CHOICE
{
      outOfService      NULL,
      maintenance       NULL,
      terminated        NULL,
      expired           NULL,
      ...
}
```

```
DescriptorRequest ::= SEQUENCE
{
      descriptorID      SEQUENCE OF DescriptorID,
      ...
}


DescriptorConfirmation ::= SEQUENCE
{
      descriptor        SEQUENCE OF Descriptor,
      ...
}


DescriptorRejection ::= SEQUENCE
{
      reason                  DescriptorRejectionReason,
      descriptorID            DescriptorID OPTIONAL,
      ...
}

DescriptorRejectionReason ::= CHOICE
{
      packetSizeExceeded      NULL, -- use other transport type
      illegalID               NULL, -- no descriptor for provided descriptorID
      security          NULL, -- request did not meet security requirements
      hopCountExceeded  NULL,
      noServiceRelationship NULL,
      undefined         NULL,
      ...
}




DescriptorIDRequest ::= SEQUENCE
{
      ...
}


DescriptorIDConfirmation ::= SEQUENCE
{
      descriptorInfo          SEQUENCE OF DescriptorInfo,
      ...
}


DescriptorIDRejection ::= SEQUENCE
{
      reason            DescriptorIDRejectionReason,
      ...
}

DescriptorIDRejectionReason ::= CHOICE
{
      noDescriptors           NULL, -- no descriptors to report
      security          NULL, -- request did not meet security requirements
      hopCountExceeded  NULL,
      noServiceRelationship   NULL,
      undefined         NULL,
      ...
}
```

```
DescriptorUpdate ::= SEQUENCE
{
      sender            AliasAddress,
      updateInfo        SEQUENCE OF UpdateInformation,
      ...
}

UpdateInformation ::=   SEQUENCE
{
      descriptorInfo    CHOICE      {
            descriptorID      DescriptorID,
            descriptor        Descriptor,
            ...
      },
      updateType  CHOICE
      {
            added             NULL,
            deleted           NULL,
            changed           NULL,
            ...
      },
      ...
}

DescriptorUpdateAck ::= SEQUENCE
{
      ...
}


AccessRequest ::= SEQUENCE
{
      destinationInfo   PartyInformation,
      sourceInfo        PartyInformation OPTIONAL,
      callInfo          CallInformation OPTIONAL,
      usageSpec         UsageSpecification OPTIONAL,  ...
}


AccessConfirmation ::= SEQUENCE
{
      templates         SEQUENCE OF AddressTemplate,
      partialResponse   BOOLEAN,
      ...
}


AccessRejection ::= SEQUENCE
{
      reason                  AccessRejectionReason,
      ...
}

AccessRejectionReason ::= CHOICE
{
      noMatch                 NULL, -- no template matched the destinationInfo
      packetSizeExceeded      NULL, -- use other transport type
      security          NULL, -- request did not meet security requirements
      hopCountExceeded  NULL,
      needCallInformation     NULL,       -- Call Information must be specified
      noServiceRelationship   NULL,
```

```
      undefined         NULL,
      ...
}

UsageRequest ::= SEQUENCE
{
      callInfo    CallInformation,
      usageSpec   UsageSpecification,
      ...
}


UsageConfirmation ::= SEQUENCE
{
      ...
}

UsageRejection ::= SEQUENCE
{
      reason                  UsageRejectReason,
      ...
}

UsageIndication ::= SEQUENCE
{
      callInfo                CallInformation,
      accessTokens            SEQUENCE OF AccessToken OPTIONAL,
      senderRole              Role,
      usageCallStatus         UsageCallStatus,
      srcInfo                 PartyInformation OPTIONAL,
      destAddress             PartyInformation,
      startTime               TimeStamp OPTIONAL,
      endTime                 TimeStamp OPTIONAL,
      terminationCause        TerminationCause OPTIONAL,
      usageFields             SEQUENCE OF UsageField,
      ...
}

UsageField ::= SEQUENCE
{
      id                      OBJECT IDENTIFIER,
      value                   OCTET STRING,
      ...
}



UsageRejectReason ::= CHOICE
{
      invalidCall             NULL,
      unavailable             NULL,
      security                NULL,
      noServiceRelationship   NULL,
      undefined               NULL,
      ...
}

UsageIndicationConfirmation ::= SEQUENCE
{
      ...
}

UsageIndicationRejection ::= SEQUENCE
{
```

```
      reason                  UsageIndicationRejectionReason,
      ...
}

UsageIndicationRejectionReason ::= CHOICE
{
      unknownCall       NULL,
      incomplete        NULL,
      security          NULL,
      noServiceRelationship   NULL,
      undefined         NULL,
      ...
}

ValidationRequest ::= SEQUENCE
{
      accessToken       SEQUENCE OF AccessToken OPTIONAL,
      destinationInfo   PartyInformation OPTIONAL,
      sourceInfo        PartyInformation OPTIONAL,
      callInfo          CallInformation,
      usageSpec         UsageSpecification OPTIONAL,
      ...
}


ValidationConfirmation ::= SEQUENCE
{
      destinationInfo   PartyInformation OPTIONAL,
      usageSpec         UsageSpecification OPTIONAL,
      ...
}


ValidationRejection ::= SEQUENCE
{
      reason                  ValidationRejectionReason,
      ...
}

ValidationRejectionReason ::= CHOICE
{
      tokenNotValid           NULL,
      security                NULL, -- request did not meet security requirements
      hopCountExceeded        NULL,
      missingSorceInfo        NULL,
      missingDestInfo         NULL,
      noServiceRelationship   NULL,
      undefined               NULL,
      ...
}

RequestInProgress ::= SEQUENCE
{
      delay       INTEGER (1..65535),
      ...
}



NonStandardRequest ::= SEQUENCE
{
      ...
}
```

```
NonStandardConfirmation ::= SEQUENCE
{
      ...
}

NonStandardRejection ::= SEQUENCE
{
      reason            NonStandardRejectionReason,
      ...
}

NonStandardRejectionReason ::= CHOICE
{
      notSupported            NULL,
      noServiceRelationship   NULL,
      undefined         NULL,
      ...
}



UnknownMessageResponse ::= SEQUENCE
{
      unknownMessage          OCTET STRING,
      reason                  UnknownMessageReason,
      ...
}

UnknownMessageReason ::= CHOICE
{
      notUnderstood           NULL,
      undefined         NULL,
      ...
}







--
-- structures common to multiple messages
--

AddressTemplate ::= SEQUENCE
{
      pattern           SEQUENCE OF Pattern,
      routeInfo         SEQUENCE OF RouteInformation,
      timeToLive        INTEGER (1..4294967295),
      ...
}

Pattern ::= CHOICE
{
      specific          AliasAddress,
      wildcard          AliasAddress,
      range       SEQUENCE          {
            startOfRange      PartyNumber,
            endOfRange  PartyNumber
      },
      ...
}
```

```
RouteInformation ::= SEQUENCE
{
      messageType CHOICE
      {
            sendAccessRequest NULL,
            sendSetup         NULL,
            nonExistent       NULL,
            ...
      },
      callSpecific      BOOLEAN,
      usageSpec   UsageSpecification OPTIONAL,
      priceInfo   SEQUENCE OF PriceInfoSpec OPTIONAL,
      contacts    SEQUENCE OF ContactInformation,
      type        EndpointType OPTIONAL,
                  -- must be present if messageType = sendSetup
      ...}
ContactInformation ::= SEQUENCE{    transportAddress  AliasAddress,     priority
            INTEGER (0..127), transportQoS      TransportQOS OPTIONAL,
      security          SEQUENCE OF SecurityMode OPTIONAL,
      accessTokens            SEQUENCE OF AccessToken OPTIONAL,
      ...
}


PriceInfoSpec ::= SEQUENCE
{
      currency                IA5String (SIZE(3)),          -- e.g. "USD"
      currencyScale                 INTEGER(-127..127),
      validFrom               GlobalTimeStamp OPTIONAL,
      validUntil              GlobalTimeStamp OPTIONAL,
      hoursFrom               IA5String (SIZE(6)) OPTIONAL, -- "HHMMSS" UTC
      hoursUntil              IA5String (SIZE(6)) OPTIONAL, -- "HHMMSS" UTC
      priceElement                  SEQUENCE OF PriceElement OPTIONAL,
      priceFormula                  IA5String (SIZE(1..2048)) OPTIONAL,
      ...
}

PriceElement ::= SEQUENCE
{
      amount                        INTEGER(0..4294967295), -- meter increment
      quantum                       INTEGER(0..4294967295), -- each or part
                                                            -- thereof
      units CHOICE
      {
            seconds                 NULL,
            packets                 NULL,
            bytes                   NULL,
            initial                 NULL,
            minimum                 NULL,
            maximum                 NULL,
            ...
      },
      ...
}



Descriptor ::= SEQUENCE
{
      descriptorInfo          DescriptorInfo,
      templates         SEQUENCE OF AddressTemplate,
      gatekeeperID            GatekeeperIdentifier OPTIONAL,
      ...
}
```

```
DescriptorInfo ::= SEQUENCE
{
      descriptorID            DescriptorID,
      lastChanged       GlobalTimeStamp,
      ...
}


AlternateBEInfo ::= SEQUENCE
{
      alternateBE       SEQUENCE OF AlternateBE,
      alternateIsPermanent    BOOLEAN,
      ...
}

AlternateBE ::= SEQUENCE
{
      contactAddress          AliasAddress,
      priority          INTEGER (1..127),
      elementIdentifier ElementIdentifier OPTIONAL,
      ...
}

AccessToken ::= CHOICE
{
      token       ClearToken,
      cryptoToken CryptoH323Token,
      ...
}

CallInformation ::= SEQUENCE
{
      callIdentifier          CallIdentifier,
      conferenceID            ConferenceIdentifier,
      ...
}


UsageCallStatus ::= CHOICE
{
      preConnect              NULL, -- Call has not started
      callInProgress                NULL, -- Call is in progress
      callEnded               NULL, -- Call ended
      ...
}

UserInformation ::= SEQUENCE
{
      userIdentifier          AliasAddress,
      userAuthenticator SEQUENCE OF CryptoH323Token OPTIONAL,
      ...
}


UsageSpecification ::= SEQUENCE
{
      sendTo                  ElementIdentifier,
      when SEQUENCE
      {
            never       NULL OPTIONAL,
            start       NULL OPTIONAL,
            end         NULL OPTIONAL,
```

```
            period          INTEGER(1..65535) OPTIONAL,   -- in seconds
            failures    NULL OPTIONAL,
            ...
      },
      required          SEQUENCE OF OBJECT IDENTIFIER,
      preferred         SEQUENCE OF OBJECT IDENTIFIER,
      ...
}


PartyInformation ::= SEQUENCE
{
      logicalAddresses  SEQUENCE OF AliasAddress,
      domainIdentifier  AliasAddress OPTIONAL,
      transportAddress  AliasAddress OPTIONAL,
      endpointType            EndpointType OPTIONAL,
      userInfo          UserInformation OPTIONAL,
      timeZone          TimeZone OPTIONAL,
      ...
}


Role ::= CHOICE
{
      originator        NULL,
      destination       NULL,
      nonStandardData   NonStandardParameter,
      ...
}

TimeZone ::=  INTEGER (-43200..43200)
                                        -- number of seconds relative to UTC
                                        -- including DST if appropriate


TerminationCause ::= SEQUENCE
{
      releaseCompleteReason         ReleaseCompleteReason,
      causeIE                       INTEGER (1..65535) OPTIONAL,
      nonStandardData               NonStandardParameter OPTIONAL,
      ...
}



AnnexGVersion           ::=   OBJECT IDENTIFIER
                        -- shall be set to
                        -- {itu-t (0) recommendation (0) h(8) h225.0(2250)
                        -- Annex (1) G (7) version (0) 1 (0)}
DescriptorID            ::=   GloballyUniqueID

ElementIdentifier ::=   BMPString (SIZE(1..128))

GlobalTimeStamp   ::=   IA5String (SIZE(14))    -- in the form YYYYMMDDHHmmSS
                                -- where YYYY = year, MM = month, DD = day,
                                -- HH = hour, mm = minute, SS = second
                                -- (for example, 19981219120000 for noon
                                -- 19 December 1998)



END   -- of ANNEXG-MESSAGES
```

Superseded by a more recent version

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	Languages and general software aspects for telecommunication systems

17130

Printed in Switzerland
Geneva, 2000