International Telecommunication Union

ITU-T H.323 System Implementors' Guide

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

(25 March 2011)

SERIES H: AUDIOVISUAL AND MULTIMEDIA SYSTEMS

Infrastructure of audiovisual services – Communication procedures

---

Implementors' Guide for Recommendations of the H.323 System (Packet-based multimedia communications systems):

H.323, H.225.0, H.245, H.246, H.283, H.341, H.450 Series, H.460 Series, and H.500 Series

ITU-T

International Telecommunication Union

Summary

This document is a compilation of reported defects identified in the versions of ITU-T Recommendation H.323 and its related Recommendations currently in force. It must be read in conjunction with the Recommendations to serve as an additional authoritative source of information for implementors. The changes, clarifications and corrections defined herein are expected to be included in future versions of affected H.323-series Recommendations.

This revision contains all updates submitted up to and including those at Study Group 16 meeting in Geneva, 14–25 March, 2011.

This Implementors' Guide was approved by ITU-T Study Group 16 on 25 March 2011 (TD 345/Plen) and it obsoletes the earlier version of this Implementors' Guide approved on 6 November 2009.

Contact Information

ITU-T Study Group 16 / Rapporteur Question 2/16	Paul E. Jones Cisco Systems, Inc. 7025 Kit Creek Road Research Triangle Park, NC 27709. USA	Tel: Fax: E-mail:	+1 919 476 2048 paulej@packetizer.com
ITU-T Study Group 16 / Rapporteur Question 3/16	Christian Groves NTEC Australia Pty. Ltd. 48 Percy St, Newport Victoria 3015. Australia	Tel: Fax: Email:	+61 3 9391 3457 +61 3 9391 3457 christian.groves@nteczone.com
ITU-T Study Group 16 / Editor ITU-T Rec. H.235.0 Editor Rec. H.235 series Implementors' Guide	Martin Euchner	E-mail:	martin.euchner@itu.int
Editor ITU-T Rec. H.323	Stephen Botzko Polycom 100 Minuteman Road Andover, MA 01810, USA	Tel: Fax: Email:	+1 978 292-5395 +1 978 292-5395 stephen.botzko@polycom.com
Editor ITU-T Rec. H.341	Craig Blasberg Cisco Systems, Inc. 7025 Kit Creek Road Research Triangle Park, NC 27709. USA	Tel: Fax: E-mail:	+1 919 392 5760x +1 919 392 6801 blasberg@cisco.com
Editor ITU-T Rec. H.225.0 Editor ITU-T Rec. H.323 Series Implementors' Guide	Muthu Arul Mozhi Perumal Cisco Systems, Inc. Divyashree Chambers 'B' Wing, No.11 O'Shaugnessey Road, Off Langford Road Bangalore, India – 560027	Tel: Fax: E-mail:	+91-80-41033563 +91-80-22230167 mperumal@cisco.com
Editor ITU-T Rec. H.225.0 Annex G	Miner Gleason Cisco Systems, Inc. 7025 Kit Creek Road Research Triangle Park, NC 27709, USA	Tel: Fax: E-mail:	+1 919 392 8752 +1 919 392 7065 mgleason@cisco.com
Editor ITU-T Rec. H.245	Mike Nilsson BT Labs Ipswitch, United Kingdom	Tel: Fax: E-mail:	+44 1 473 645413 +44 1 473 643791 mike.nilsson@bt.com
Editor ITU-T Rec. H.246	Patrick Luthi Cisco Systems Norway, Norway	Tel: E-mail:	+47 67 125 125 patrick.luthi@cisco.com
Editor ITU-T Rec. H.450.7	Dave Walker SS8 Networks 135 Michael Cowpland Drive, Suite 200 Kanata, Ontario, K2M 2E9, Canada	Tel: Fax: E-mail:	+1 613 592 8450 +1 613 592 9634 drwalker@rogers.com
Editor ITU-T Rec. H.450.8	Glen Freundlich Avaya Communication 1300 W. 120th Avenue Westminster, CO 80234, USA	Tel: Fax: E-mail:	+1 303 538 2899 +1 303 538 3007 ggf@avaya.com
Editor ITU-T Rec. H.460.1	P. Cordell	E-mail:	pete@tech-know-ware.com
Editor ITU-T Rec. H.460.4	Gary Thom	E-mail:	gthom@delta-info.com
Editor ITU-T Rec. H.460.5	Sasha Ruditsky	E-mail:	sasha@radvision.com
Editor ITU-T Rec. H.460.6	Bob Gilman	E-mail:	bob_gilman@comcast.net
Editor ITU-T Rec. H.460.{2,7,8}	Paul Jones (see above for Q2/16)	E-mail:	paulej@packetizer.com
Editor ITU-T Rec. H.460.3	Louis Fourie	E-mail:	lfourie@cisco.com
Editor ITU-T Rec. H.460.9	Ernst Horvath Siemens Austria Gudrunstrasse 11 A-1101 Vienna. Austria	Tel: Fax: E-mail:	+43 5 1707 45897 +43 5 1707 56992 ernst.horvath@siemens-enterprise.com

Note: Not all Recommendations indicated above have IG issues in this document. The information above is provided for completeness.

Table of Contents

IMPLEMENTORS' GUIDE FOR RECOMMENDATIONS OF THE H.323 SYSTEM (PACKET-BASED MULTIMEDIA COMMUNICATIONS SYSTEMS)

1 Scope

This guide resolves defects in the following categories:

- editorial errors
- technical errors, such as omissions and inconsistencies
- ambiguities

In addition, the Implementors' Guide may include explanatory text found necessary as a result of interpretation difficulties apparent from the defect reports.

This Guide will not address proposed additions, deletions, or modifications to the Recommendations that are not strictly related to implementation difficulties in the above categories. Proposals for new features should be made in through contributions to the ITU-T.

2 Introduction

This document is a compilation of reported defects identified in the versions of ITU-T Recommendation H.323 and its related Recommendations currently in force. It must be read in conjunction with the Recommendations to serve as an additional authoritative source of information for implementors. The changes, clarifications and corrections defined herein are expected to be included in future versions of affected H.323-series Recommendations.

Upon discovering technical defects with any components of the H.323 Recommendations series, please provide a written description directly to the editors of the affected Recommendations with a copy to the Q2/16 Rapporteur. The template for a defect report is located at the end of the Guide. Contact information for these parties is included at the front of the document. Return contact information should also be supplied so a dialogue can be established to resolve the matter and an appropriate reply to the defect report can be conveyed. This defect resolution process is open to any interested party. Formal membership in the ITU is not required to participate in this process.

3 References

This document refers to the following ITU-T Recommendations:

- ITU-T Recommendation H.323 (2006), Packet-Based multimedia communications systems
- ITU-T Recommendation H.225.0 (2006), Call signaling protocols and media stream packetization for packet based multimedia communications Systems
- ITU-T Recommendation H.245 (5/2006), Control protocol for multimedia communication
- ITU-T Recommendation H.246 (2006), Interworking of H-Series multimedia terminals with H-Series multimedia terminals and voice/voiceband terminals on GSTN and ISDN
- ITU-T Recommendation H.235.0 – H.235.9 (2005), Security and encryption for H Series (H.323 and other H.245 based) multimedia terminals
- ITU-T Recommendation H.450.1 (2011), Generic functional protocol for the support of supplementary services in H.323

- ITU-T Recommendation H.450.2 (2011), Call transfer supplementary service for H.323
- ITU-T Recommendation H.450.3 (2011), Call diversion supplementary service for H.323
- ITU-T Recommendation H.450.4 (1999), Call hold supplementary service for H.323
- ITU-T Recommendation H.450.5 (1999), Call park and call pickup supplementary services for H.323
- ITU-T Recommendation H.450.6 (1999), Call waiting supplementary service for H.323
- ITU-T Recommendation H.450.7 (1999), Message waiting indication supplementary service for H.323
- ITU-T Recommendation H.450.8 (2000), Name identification supplementary service for H.323
- ITU-T Recommendation H.450.9 (2000), Call Completion Supplementary Services for H.323
- ITU-T Recommendation H.450.10 (2001), Call offer supplementary service for H.323
- ITU-T Recommendation H.450.11 (2001), Call intrusion supplementary services
- ITU-T Recommendation H.450.12 (2001), Call Information Additional Network Feature for H.323
- ITU-T Recommendation H.460.1 (2002), Guidelines for the use of generic extensibility framework
- ITU-T Recommendation H.460.2 (2001), Number Portability interworking between H.323 and SCN networks
- ITU-T Recommendation H.460.3 (2002), Circuit status map within H.323 systems
- ITU-T Recommendation H.460.4 (2002), Call priority designation for H.323 calls
- ITU-T Recommendation H.460.5 (2002), H.225.0 transport of multiple Q.931 IE of the same type
- ITU-T Recommendation H.460.6 (2002), Extended Fast Connect Feature
- ITU-T Recommendation H.460.7 (2002), Digit Maps Within H.323 Systems
- ITU-T Recommendation H.460.8 (2002), Querying for alternate routes within H.323 systems
- ITU-T Recommendation H.460.9 (2002), Support for online QoS-Monitoring report
- ITU-T Recommendation H.460.10 (2004), Call party category within H.323 systems
- ITU-T Recommendation H.460.11 (2004), Delayed call establishment within H.323 systems
- ITU-T Recommendation H.460.12 (2004), Glare control indicator within H.323 systems
- ITU-T Recommendation H.460.13 (2004), Called user release control
- ITU-T Recommendation H.460.14 (2004), Support for Multi-Level Precedence and Preemption (MLPP) within H.323 systems
- ITU-T Recommendation H.460.15 (2004), Call signalling transport channel suspension and redirection within H.323 systems

– ITU-T Recommendation H.460.16 (2005), Multiple message release sequence capability

– ITU-T Recommendation H.460.17 (2005), Using H.225.0 call signalling connection as transport for H.323 RAS messages

– ITU-T Recommendation H.460.18 (2005), Traversal of H.323 signalling across network address translators and firewalls

– ITU-T Recommendation H.460.19 (2005), Traversal of H.323 media across network address translators and firewalls

– ITU-T Recommendation H.460.20 (2005), Location number within H.323 systems

– ISO/IEC 11571 (1998), Information technology – Telecommunications and information exchange between systems – Private Integrated Services Networks – Addressing

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

– ITU-T Recommendation H.283, Remote device control logical channel transport

4 Nomenclature

In addition to traditional revision marks, the following marks and symbols are used to indicate to the reader how changes to the text of a Recommendation should be applied:

Symbol	Description
[Begin Correction]	Identifies the start of revision marked text based on extractions from the published Recommendations affected by the correction being described.
[End Correction]	Identifies the end of revision marked text based on extractions from the published Recommendations affected by the correction being described.
...	Indicates that the portion of the Recommendation between the text appearing before and after this symbol has remained unaffected by the correction being described and has been omitted for brevity.
--- SPECIAL INSTRUCTIONS --- {instructions}	Indicates a set of special editing instructions to be followed.

5 Technical and Editorial Corrections to H.323 Series Recommendations

5.1 Technical and Editorial Corrections to ITU-T Recommendation H.323 (2009)

None for this version of the H.323 System IG.

5.2 Technical and Editorial Corrections to ITU-T Recommendation H.225.0 (2009)

None for this version of the H.323 System IG.

5.3 Technical and Editorial Corrections to ITU-T Recommendation H.245 (2009)

None for this version of the H.323 System IG.

5.4 Technical and Editorial Corrections to ITU-T Recommendation H.246 (2006)

None for this version of the H.323 System IG.

5.5 Technical and Editorial Corrections to ITU-T Recommendation H.235 Series

Corrections to H.235 series Recommendations are specified in H.235 Series Implementors' Guide.

5.6 Technical and Editorial Corrections to ITU-T Recommendation H.450 Series

5.6.1 Technical and Editorial Corrections to H.450.4 (1999)

5.6.1.1 Change Relating to Interpretation APDU

Description:	In order to align H.450.4 with other H.450-series A modified description of the Call Hold Interpretation APDU (i-apdu) setting has been added in clause 6 of Recommendation H.450.4. This information will be contained in the revision 2 of H.450.4 Recommendation to be published by the ITU-T. The modified text is shown below.

[Begin Correction]

6 Messages and Information elements

...

When conveying the Invoke APDU of operations **remoteHold** and **remoteRetrieve**, the Interpretation APDU shall be omitted or shall contain the value **rejectAnyUnrecognizedInvokePdu**.

[End Correction]

5.6.1.2 Feature Interaction between H.450.4 and H.450.2

Description:	A modified description of the Call Hold interaction with Call Transfer has been added in clause 9.2.1 of Recommendation H.450.4. This information will be contained in the revision 2 of H.450.4 Recommendation to be published by the ITU-T. The modified text is shown below.

[Begin Correction]

9.2.1 Call Transfer (H.450.2)

If prior to Consultation, the first call has been put on hold, the served User endpoint shall decide whether or not to automatically retrieve the held User before Call Transfer is invoked.

- If the served User endpoint decides for the automatic retrieve option, a~~A~~ **retrieveNotific** Invoke APDU (in case of near end call hold) or a **remoteRetrieve** Invoke APDU (in case of remote-end call hold) may either be sent by the served user prior to the message containing the **callTransferInitiate** Invoke APDU or may be sent within the same message containing the **callTransferInitiate** Invoke APDU.

  If call transfer fails after retrieval from hold was successful (i.e. if callTransferInitiate Return Error or Reject APDU is received or if timer CT-T3 expires), the served user endpoint may automatically re-invoke SS-Hold.

  If remote-end call hold retrieval is unsuccessful, in order to proceed with call transfer the remoteRetrieve Return Error or remoteRetrieve Reject APDU should be disregarded.

- If the served User endpoint decides to not choose the automatic retrieve option, call hold applies to the primary call until call transfer has been completed successfully (i.e. until the primary call is cleared). If transfer fails, the primary call remains being held by User A.

[End Correction]

5.6.1.3 Correction to Section 11/H.450.4 Title

Description:	Section 11/H.450.4 describes both near-end and remote-end call-hold. The section title is corrected so that is does not refer to near-end hold alone. This information will be contained in the revision 2 of H.450.4 Recommendation to be published by the ITU-T. The modified text is shown below.

[Begin Correction]

11 Dynamic description for ~~near-end~~ call hold

11.1 Operational models and signalling flows

11.1.1 Near-end call hold

[End Correction]

5.6.1.4 Corrections relating to Held State in text and Remote Hold SDLs

Description:	(i) Hold_RE_Held state is used to indicate successful call hold state on both holding and held endpoints. This can be confusing, especially when reading the SDL. New Hold_xx_Holding states, that indicate successful call-hold, are introduced for near-end and remote-end hold procedures. (ii) First diagram in Figure 18/H.450.4 indicates that if the served endpoint sends a remoteHold.req primitive then the state changes to Hold_RE_Retrieve_Req even though call retrieval has not been requested. The state should remain unchanged instead.

	(iii) Second diagram in Figure 18/H.450.4 incorrectly mentions remoteRetrieve.rr instead of remoteRetrieve.re (ReturnResult instead of ReturnError). This information will be contained in the revision 2 of H.450.4 Recommendation to be published by the ITU-T. The modified text is shown below.

[Begin Correction]

7.1.1 Near-end call hold

On receiving a near-end call hold request from the local User when SS-HOLD is allowed, the holding endpoint shall send a FACILITY message with a **holdNotific** Invoke APDU to the remote endpoint and shall enter the Hold_NE_Holding~~eld~~ state. MOH shall be provided to the held user.

On receiving a near-end retrieve request from the local User, the holding endpoint shall check whether the call to which the retrieve request applies is in state Hold_NE_~~Held~~Holding. If so, the holding endpoint shall send a FACILITY message with a **retrieveNotific** Invoke APDU to the held endpoint and shall stop sending MOH to the remote endpoint. User A and User B may then continue communicating with each other.

…

7.1.2 Remote-end call hold

…

On receipt of a FACILITY message with a **remoteHold** Return Result APDU, timer T1 shall be stopped and the Hold_RE_~~Held~~ Holding state shall be entered.

…

Upon receiving a remote retrieve request from the local User, the served endpoint shall check whether the call to which the retrieve request applies is in state Hold_RE_~~Held~~Holding. If so, the served endpoint shall send a FACILITY message with a **remoteRetrieve** Invoke APDU to the held endpoint, start timer T2 and enter state Hold_RE_Retrieve_Req.

…

7.2.1 Near-end call hold

A Reject APDU received as a response to a **holdNotific** Invoke APDU or as a response to **retrieveNotific** Invoke APDU while in state Hold_NE_~~Held~~ Holding shall be ignored (meaning that the remote endpoint does not understand the SS-HOLD supplementary service). In such cases, the held user B knows about the hold condition only by the reception of MOH and about the retrieval condition by the cessation of MOH and the resumption of normal communications.

7.2.2 Remote-end call hold

The holding endpoint shall not allow the local User to invoke multiple, simultaneous remote-hold requests for the same call. That is, the holding endpoint shall not send a **remoteHold** Invoke APDU for a call:

1) while timer T1 is running for that call; or

2) when that call is already in the Hold_RE_Holding~~eld~~ state.

…

11.2.4 States

Hold_Idle No call hold procedure has been initiated.

Hold_NE_~~Held~~Holding NN~~N~~ear-end call hold has been invoked by holding endpoint.

Hold_RE_Requested Remote-end call hold has been requested, waiting for response.

Hold_RE_Holding Remote-end call hold has been invoked successfully.

~~Hold_RE_Held Remote-end call hold has been requested successfully.~~

Hold_RE_Retrieve_Req Remote-end call hold retrieval has been requested.

…

12 Operations in support of SS-call hold

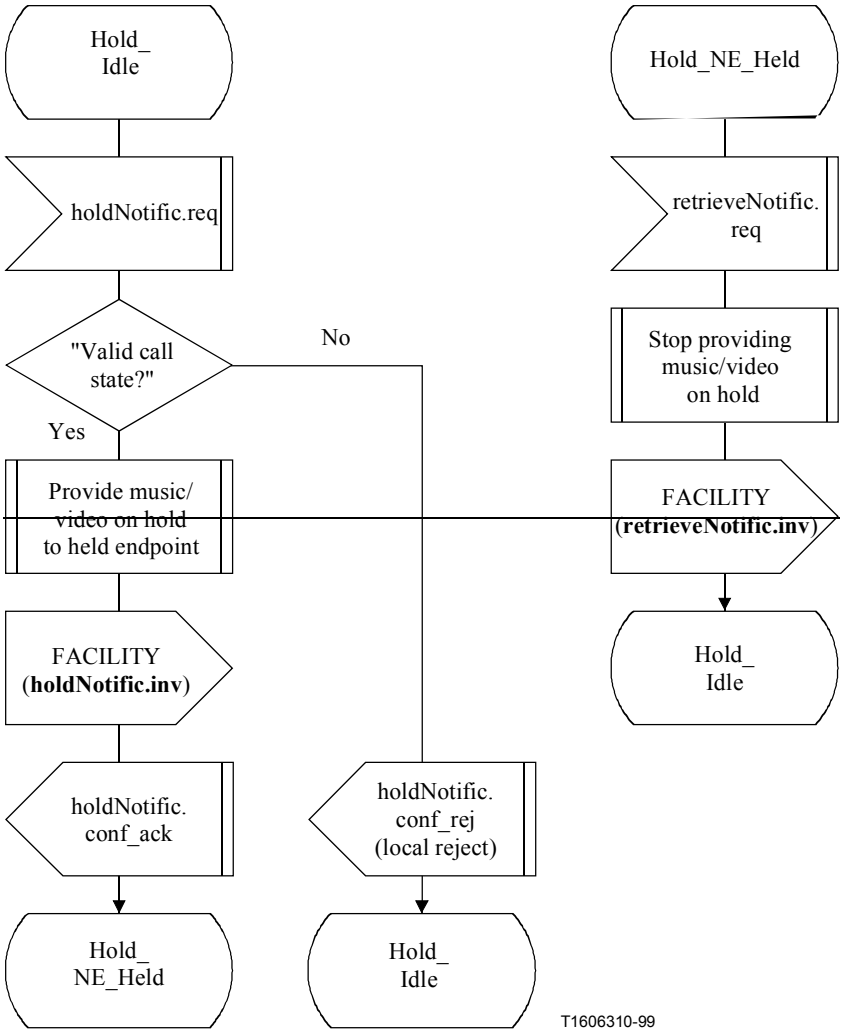
…

```
remoteRetrieve          OPERATION ::=
    { -- sent from holding to held H.323 endpoint to invoke retrieval of
remote-end call hold feature
    ARGUMENT  RemoteRetrieveArg  OPTIONAL TRUE
    RESULT          RemoteRetrieveRes  OPTIONAL TRUE
    ERRORS {invalidCallState |
                -- Call to which retrieve request applies is not in
state Hold_RE_Holding~~eld~~
              undefined -- undefined reason
              }
    CODE local: 104
    }
```

…

13.1 Near-end call hold SDLs

See Figures 14 and 15.


Hold_
Idle
holdNotific.req
"Valid call
state?"
No
Yes
Provide music/
video on hold
to held endpoint
FACILITY
(holdNotific.inv)
holdNotific.
conf_ack
Hold_
NE_Held
holdNotific.
conf_rej
(local reject)
Hold_
Idle
Hold_NE_Held
retrieveNotific.
req
Stop providing
music/video
on hold
FACILITY
(retrieveNotific.inv)
Hold_
Idle
T1606310-99

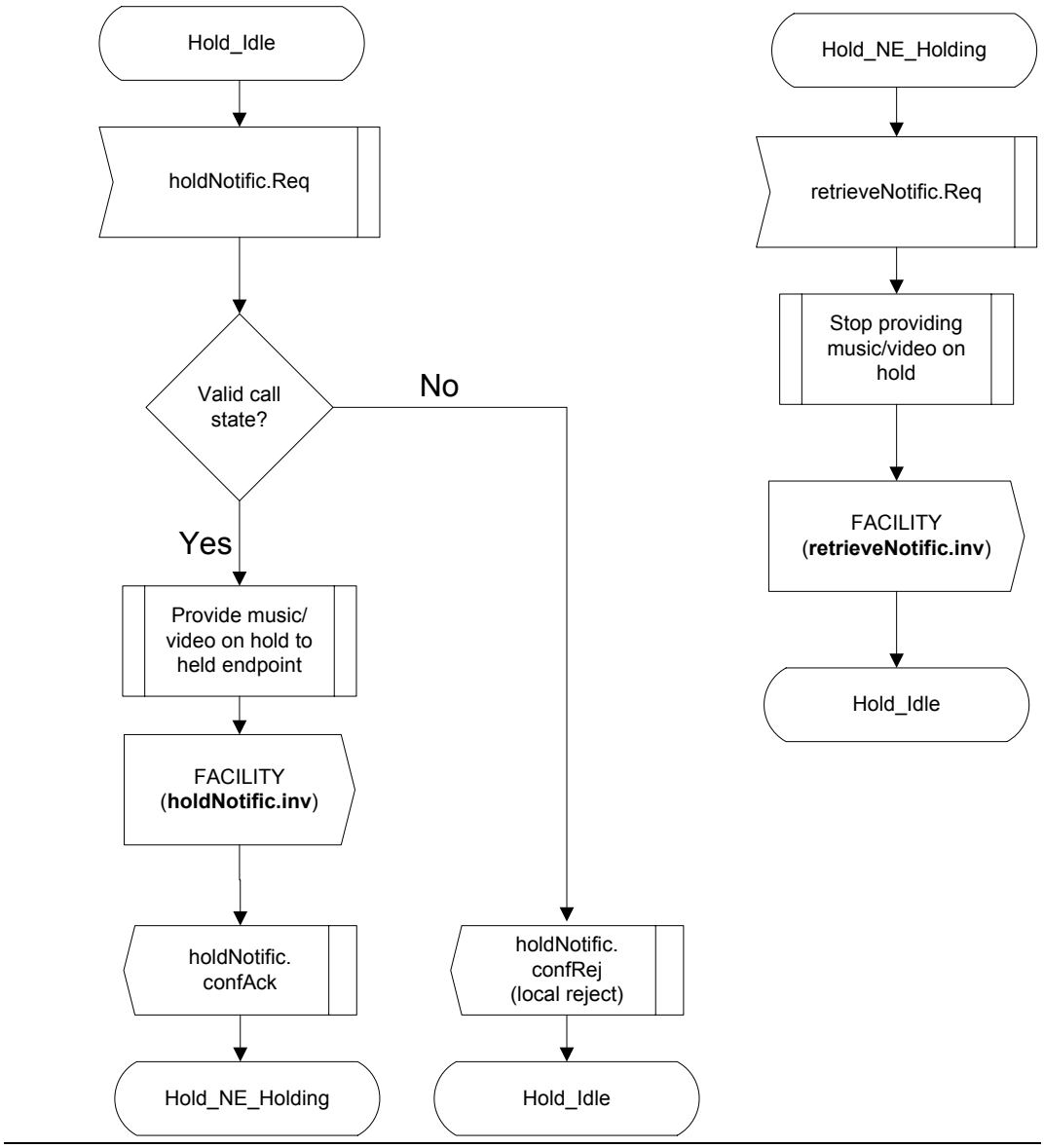


Figure 14/H.450.4 – Near-end call hold – Holding SDL

…

13.2 Remote-end call hold SDLs

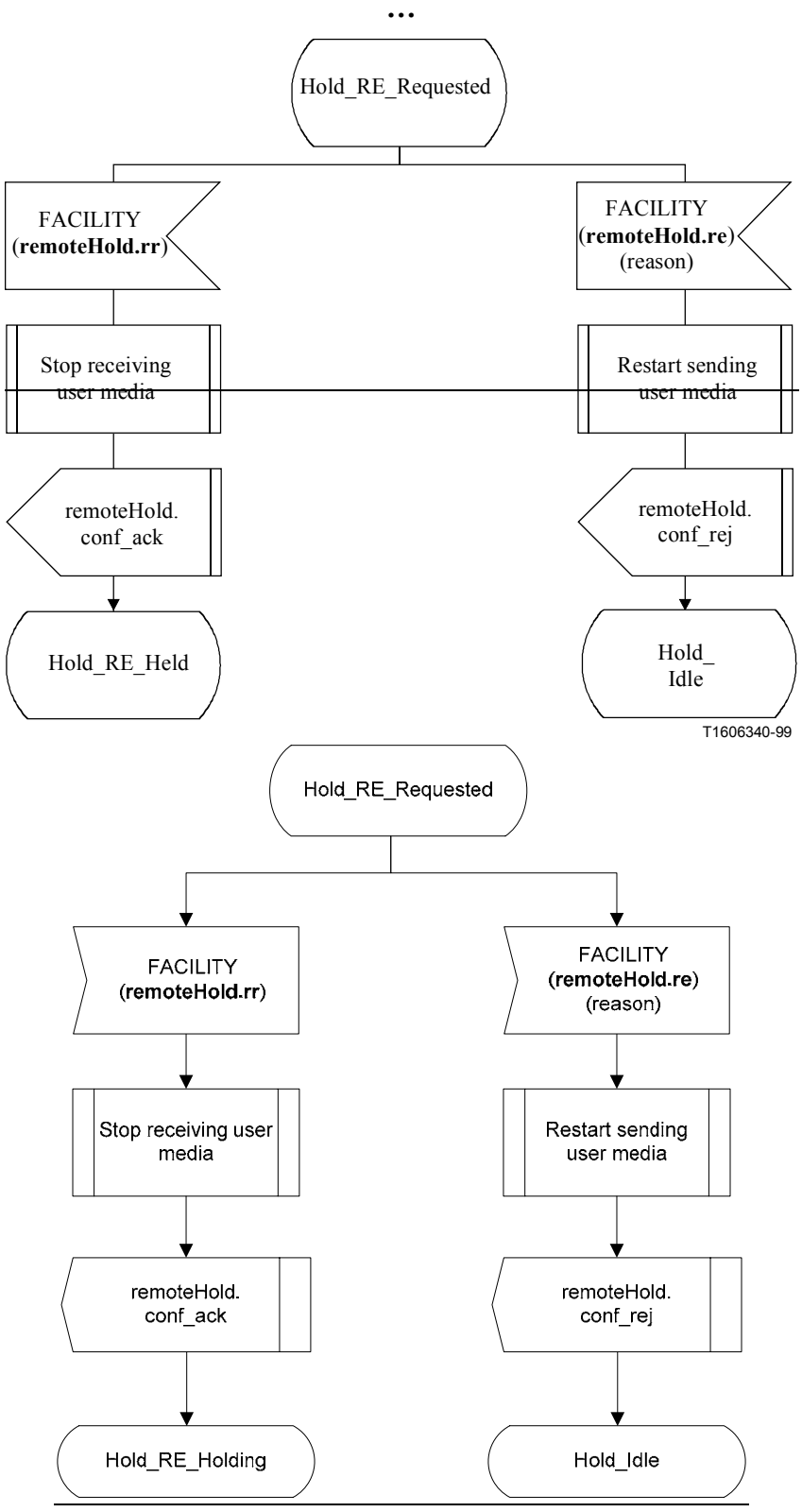
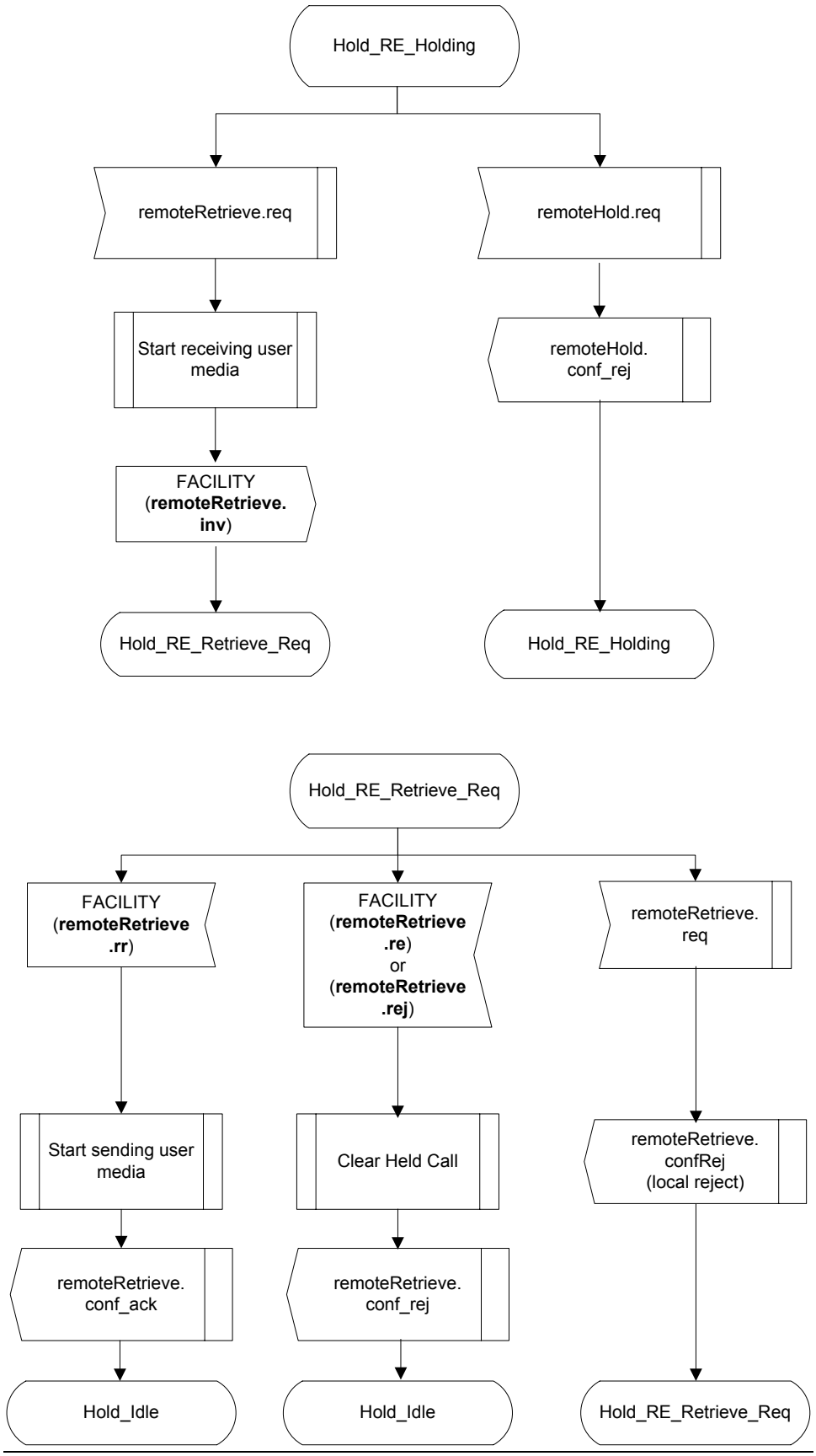


Figure 17/H.450.4 – Remote-end call hold – Holding SDL *(sheet 2 of 3)*


Hold_RE_Holding
remoteRetrieve.req
remoteHold.req
Start receiving user media
remoteHold. conf_rej
FACILITY (remoteRetrieve. inv)
Hold_RE_Retrieve_Req
Hold_RE_Holding
Hold_RE_Retrieve_Req
FACILITY (remoteRetrieve .rr)
FACILITY (remoteRetrieve .re) or (remoteRetrieve .rej)
remoteRetrieve. req
Start sending user media
Clear Held Call
remoteRetrieve. confRej (local reject)
remoteRetrieve. conf_ack
remoteRetrieve. conf_rej
Hold_Idle
Hold_Idle
Hold_RE_Retrieve_Req

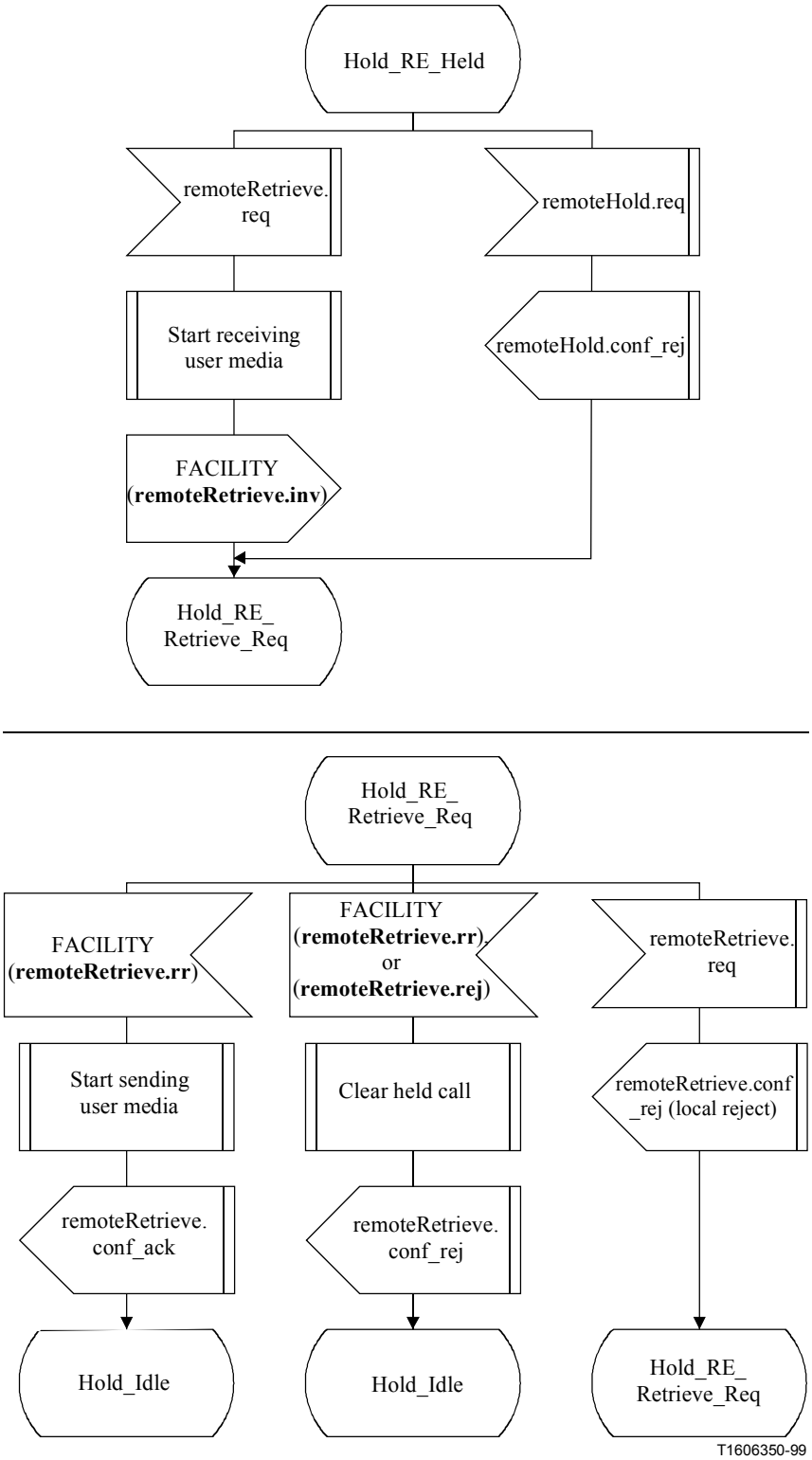


Figure 18/H.450.4 – Remote-end call hold – Holding SDL *(sheet 3 of 3)*

[End Correction]

5.6.2 Technical and Editorial Corrections to H.450.5 (1999)

5.6.2.1 Clarification of the CallIdentifier

Description:	A clarification of the setting of H.225.0 element CallIdentifier in conjunction with H.450.5 parked calls has been added within clause 8.3 "Interactions with H.225.0 parameters". This information will be contained in the revision 2 of H.450.5 Recommendation to be published by the ITU-T. The modified text is shown below.

[Begin Correction]

8.3 Interaction with H.225.0 parameters

The H.225.0 **CallIdentifier** value within a parked call shall use a new value, rather~~be set to~~ the CallIdentifier value that was used in the primary call. For all other SETUP messages carrying SS-PARK or SS-PICKUP related APDUs as defined within this R~~r~~ecommendation, new CallIdentifier values shall be used. Note that the CallIdentifier value of the parked/alerting call is preserved during the SS-PARK / SS-PICKUP procedure within the H.450 APDUs.

[End Correction]

5.6.3 Technical and Editorial Corrections to H.450.7 (1999)

5.6.3.1 Change Relating to Interpretation APDU

Description:	In order to align H.450.7 with other H.450-series, a modified description of the Message Waiting Indication Interpretation APDU (i-apdu) setting has been added in clause 7.1.1 of Recommendation H.450.7. This information will be contained in the revision 2 of H.450.7 Recommendation to be published by the ITU-T. The modified text is shown below.

[Begin Correction]

7.1.1 H.450.1 Supplementary Service APDU

...

When conveying the Invoke APDU of operations **mwiActivate**, **mwiDeactivate**, and **mwiInterrogate**, the interpretation APDU shall be omitted or shall contain the value **rejectAnyUnrecognizedInvokePdu**. ~~This is implicitly equivalent to specifying an interpretation APDU of rejectAnyUnrecognizedInvokePDU.~~

[End Correction]

5.6.4 Technical and Editorial Corrections to H.450.8 (2000)

5.6.4.1 Usage of CalledName and AlertingName

Description:	An editorial error has been found in the H.450.8 (2000) Recommendation in the usage of calledName and alteringName. The following text corrects the errors.

[Begin Correction]

7.2 Terminals or MCU as Originating Endpoint

…

A terminal or MCU in receipt of an H.225.0 Connect, Alerting, or Release Complete message containing a connectedName, ~~called~~alertingName, or busyName APDU should not present name information if the Name element indicates namePresentationRestricted.

8.2 Terminals or MCU as Terminating Endpoint

A terminal or MCU in receipt of the H.225.0 Setup message may include name information in the Connect, Alerting or Release Complete as described above in 6.2, 6.3 or 6.4. If presentation of the name to the calling party is desirable, the Name element in the alertingName, connectedName, or busyName operation should indicate namePresentationAllowed. If presentation of the name to the called party is to be restricted, the Name element in the ~~called~~alertingName, connectedName, or busyName operation should indicate namePresentationRestricted.

[End Correction]

5.6.5 Technical and Editorial Corrections to H.450.12 (2001)

5.6.5.1 Technical Correction

Description:	The receipt of a CmnInform APDU at User A's Endpoint is not described. Therefore add the text below at the end of section 7.1.1.1 ANF-CMN invocation.

[Begin Correction]

7.1.1.1 ANF-CMN invocation

...

Upon receipt of a CmnInform invoke APDU in any message, the Originating endpoint shall remain in the current state.

[End Correction]

5.6.5.2 Add definition of the states CMN-Wait-Response and CMN-Wait-Answer-Response

Description:	The states CMN-Wait-Response and CMN-Wait-Answer-Response are used only in the SDL diagrams but are not defined anywhere. To avoid confusion, a definition of their meaning is added in section 13.

[Begin Correction]

13. Specification and Description Language (SDL) Diagrams for ANF-CMN

...

In the following SDLs the states CMN-Wait-Response and CMN-Wait-Answer-Response are used to describe the behavior of the Endpoints using explicit primitive exchange.

The state CMN-Wait-Response is entered at the Endpoint after a primitive CMNRequest indication is received and the previous state was CMN-Idle.

The state CMN-Wait-Answer-Response is entered at the Endpoint after a primitive CMNRequest indication is received and the previous state was CMN-Wait-Answer.

[End Correction]

5.6.5.3 Redesign the SDL Diagrams, add two missing collision branches and delete an erroneous message symbol

Description:	Two collision branches are missing: add in section 13.1 Figure 8/H.450.12 the possible receipt of a CMNInform request from the application in state CMN-Wait-Answer and in Figure 9/H.450.12 the possible receipt of a CMNRequest request in state CMN-Wait-Response. In Figure 9/H.450.12 the receipt of a CMNInform Request in state CMN-Wait-Response shall be ignored and the message with CMNInform invoke APDU shall not be forwarded to endpoint B.

[Begin Correction]

Editorial - Replace the indicated diagrams by the following:

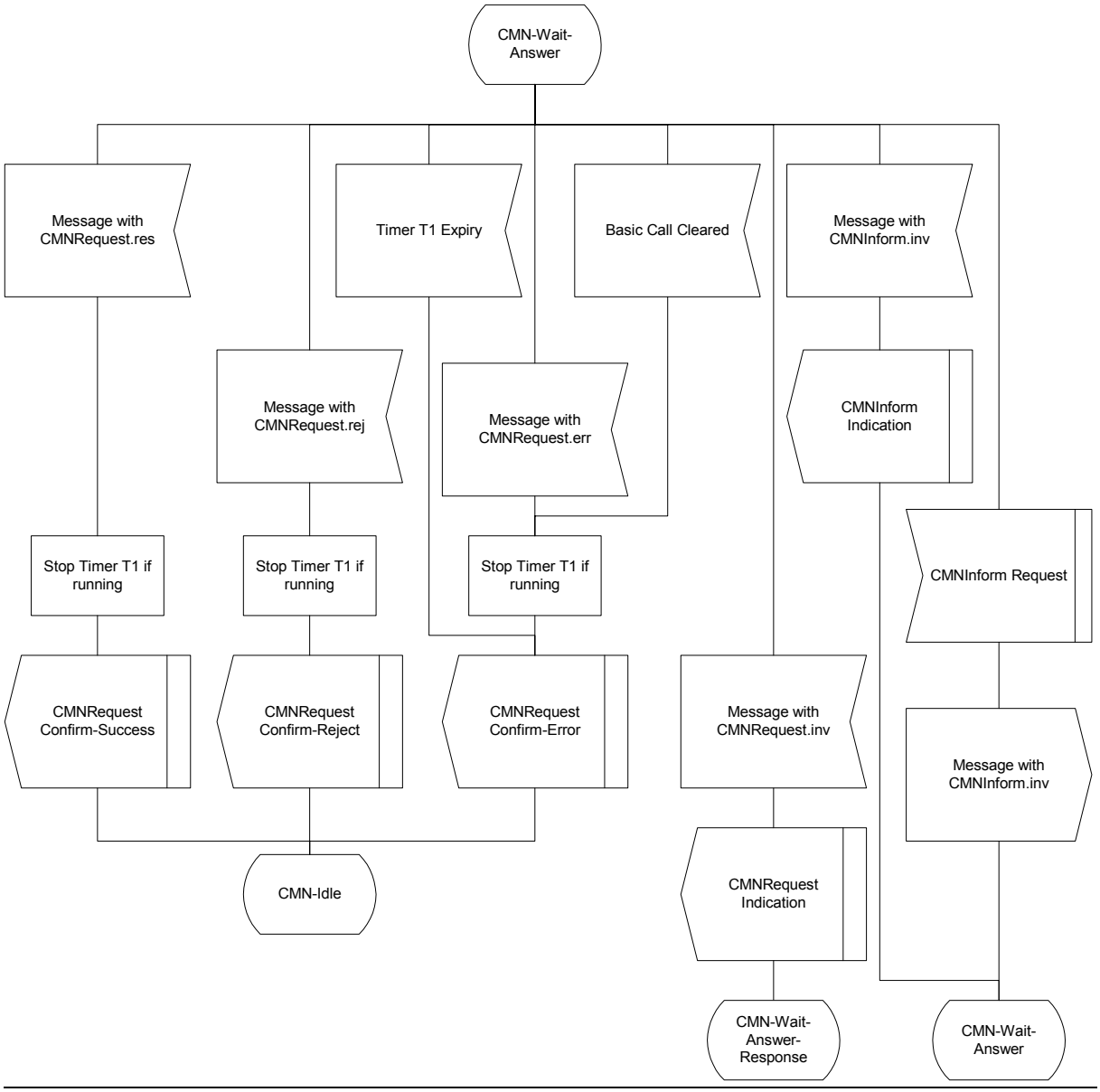


Figure 8/H.450.12 – SDL Representation of ANF-CMN at Endpoint A (Part 3)

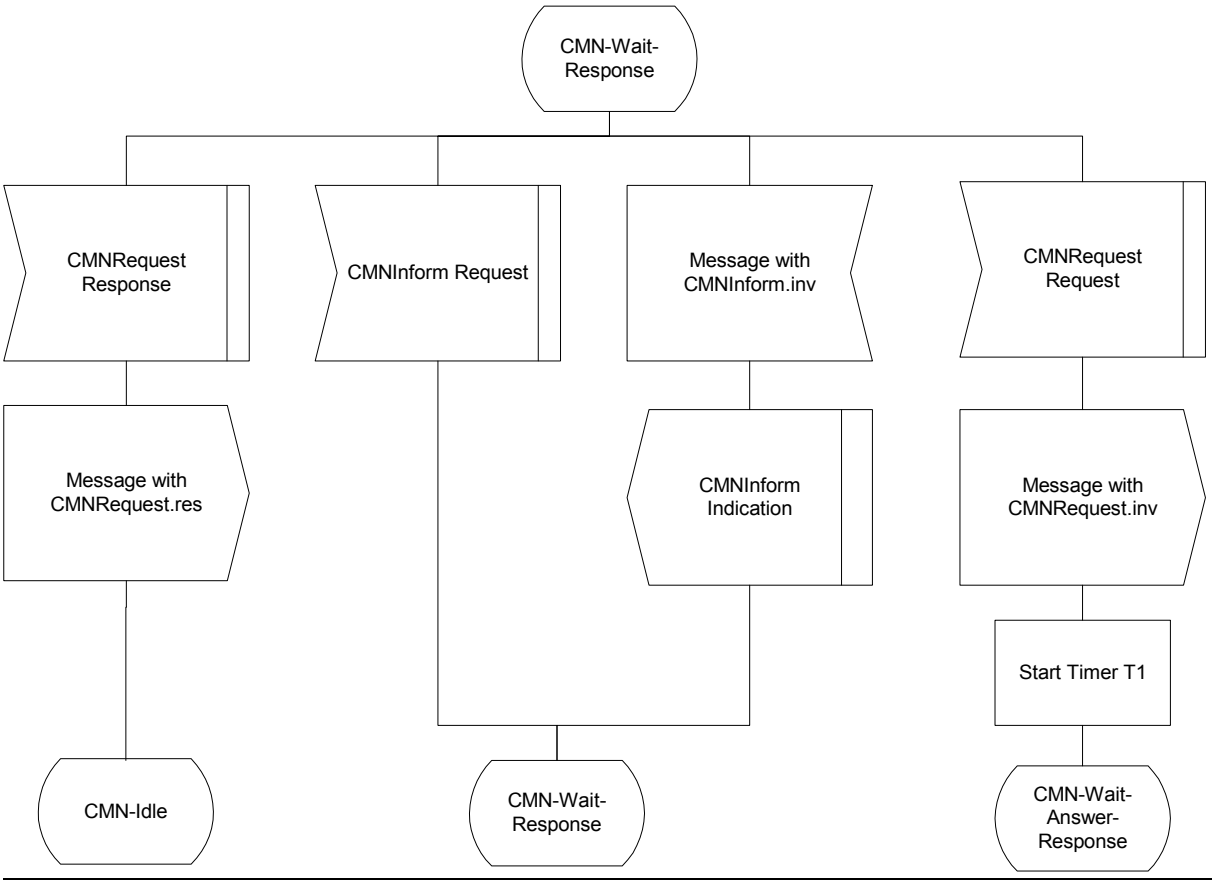


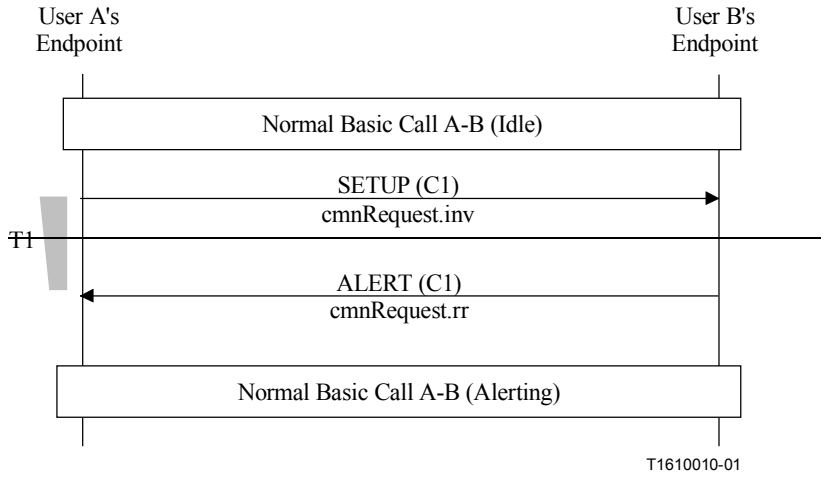
Figure 9/H.450.12 – SDL Representation of ANF-CMN at Endpoint A (Part 3)

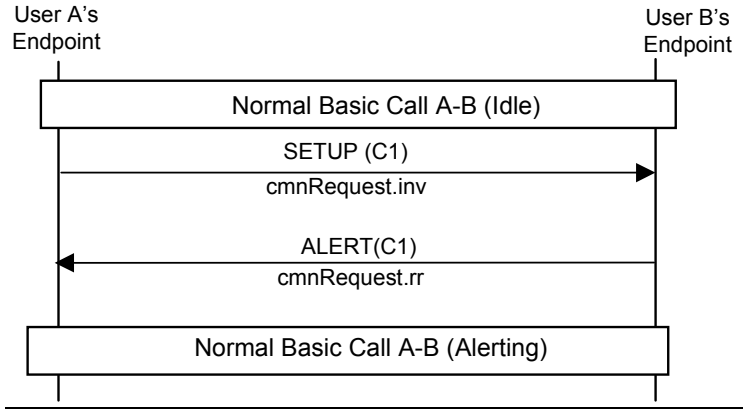
[End Correction]

5.6.5.4 Message Flow of ANF-CMN

Description:	Timer T1 is started if **cmnRequest** invoke is sent in FACILITY message, but not if it is sent in a SETUP message. However, the message flow diagram in Figure 2/H.450.12 erroneously contains timer T1. The erroneous diagram should be replaced by the corrected diagram as below.

[Begin Correction]





[End Correction]

5.7 Technical and Editorial Corrections to ITU-T Recommendation H.341 (1999)

5.7.1 Corrections to H.341 Annex B-1 H225-MIB

Description:	Each field in **CallSignalStatsEntry** SEQUENCE referred to the number of messages received ("In") and the number of messages transmitted ("Out"). These counters shall be combined.

[Begin Correction]

```
CallSignalStatsEntry::= SEQUENCE {
              callSignalStatsCallConnectionsIn
                  Counter32,
              callSignalStatsCallConnectionsOut
                  Counter32,
              callSignalStatsAlertingMsgsIn
```

```
        Counter32,
    callSignalStatsAlertingMsgsOut
        Counter32,
    callSignalStatsCallProceedingsIn
        Counter32,
    callSignalStatsCallProceedingsOut
        Counter32,
    callSignalStatsSetupMsgsIn
        Counter32,
    callSignalStatsSetupMsgsOut
        Counter32,
    callSignalStatsSetupAckMsgsIn
        Counter32,
    callSignalStatsSetupAckMsgsOut
        Counter32,
    callSignalStatsProgressMsgsIn
        Counter32,
    callSignalStatsProgressMsgsOut
        Counter32,
    callSignalStatsReleaseCompleteMsgsIn
        Counter32,
    callSignalStatsReleaseCompleteMsgsOut
        Counter32,
    callSignalStatsStatusMsgsIn
        Counter32,
    callSignalStatsStatusMsgsOut
        Counter32,
    callSignalStatsStatusInquiryMsgsIn
        Counter32,
    callSignalStatsStatusInquiryMsgsOut
        Counter32,
    callSignalStatsFacilityMsgsIn
        Counter32,
    callSignalStatsFacilityMsgsOut
        Counter32,
    callSignalStatsInfoMsgsIn
        Counter32,
    callSignalStatsInfoMsgsOut
        Counter32,
    callSignalStatsNotifyMsgsIn
        Counter32,
    callSignalStatsNotifyMsgsOut
        Counter32,
    callSignalStatsAverageCallDuration
         Integer32,
    callSignalStatsCallConnections
         Counter32,
    callSignalStatsAlertingMsgs
         Counter32,
    callSignalStatsCallProceedings
         Counter32,
    callSignalStatsSetupMsgs
         Counter32,
    callSignalStatsSetupAckMsgs
         Counter32,
    callSignalStatsProgressMsgs
         Counter32,
    callSignalStatsReleaseCompleteMsgs
         Counter32,
    callSignalStatsStatusMsgs
         Counter32,
```

```
            callSignalStatsStatusInquiryMsgs
                Counter32,
            callSignalStatsFacilityMsgs
                Counter32,
            callSignalStatsInfoMsgs
                Counter32,
            callSignalStatsNotifyMsgs
                Counter32
            }

    callSignalStatsCallConnectionsIn OBJECT-TYPE
        SYNTAX      Counter32
        MAX-ACCESS  read-only
        STATUS      deprecated~~current~~
        DESCRIPTION
            "The number of successful connections in which this entity
             has been a callee."
        ::= { callSignalStatsEntry 1 }
    callSignalStatsCallConnectionsOut OBJECT-TYPE
        SYNTAX      Counter32
        MAX-ACCESS  read-only
        STATUS      deprecated~~current~~
        DESCRIPTION
            "The number of successful connections in which this entity
             has been a caller."
        ::= { callSignalStatsEntry 2 }
    callSignalStatsAlertingMsgsIn OBJECT-TYPE
        SYNTAX      Counter32
        MAX-ACCESS  read-only
        STATUS      deprecated~~current~~
        DESCRIPTION
            "The number of alerting messages received by this entity."
        ::= { callSignalStatsEntry 3 }
    callSignalStatsAlertingMsgsOut OBJECT-TYPE
        SYNTAX      Counter32
        MAX-ACCESS  read-only
        STATUS      ~~current~~deprecated
        DESCRIPTION
            "The number of alerting messages sent by this entity."
        ::= { callSignalStatsEntry 4 }
    callSignalStatsCallProceedingsIn OBJECT-TYPE
        SYNTAX      Counter32
        MAX-ACCESS  read-only
        STATUS      ~~current~~deprecated
        DESCRIPTION
            "The number of call proceeding messages received by this
             entity."
        ::= { callSignalStatsEntry 5 }
    callSignalStatsCallProceedingsOut OBJECT-TYPE
        SYNTAX      Counter32
        MAX-ACCESS  read-only
        STATUS      ~~current~~deprecated
        DESCRIPTION
            "The number of call proceeding messages sent by this entity."
        ::= { callSignalStatsEntry 6 }


    callSignalStatsSetupMsgsIn OBJECT-TYPE
        SYNTAX      Counter32
        MAX-ACCESS  read-only
        STATUS      ~~current~~deprecated
```

```
    DESCRIPTION
        "The number of setup messages received by this entity."
    ::= { callSignalStatsEntry 7 }
callSignalStatsSetupMsgsOut OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      ~~current~~deprecated
    DESCRIPTION
        "The number of setup messages sent by this entity."
    ::= { callSignalStatsEntry 8 }
callSignalStatsSetupAckMsgsIn OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      ~~current~~deprecated
    DESCRIPTION
        "The number of setupAck messages received by this entity."
    ::= { callSignalStatsEntry 9 }
callSignalStatsSetupAckMsgsOut OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      ~~current~~deprecated
    DESCRIPTION
        "The number of setupAck messages sent by this entity."
    ::= { callSignalStatsEntry 10 }
callSignalStatsProgressMsgsIn OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      ~~current~~deprecated
    DESCRIPTION
        "The number of progress messages received by this entity."
    ::= { callSignalStatsEntry 11 }
callSignalStatsProgressMsgsOut OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      ~~current~~deprecated
    DESCRIPTION
        "The number of progress messages sent by this entity."
    ::= { callSignalStatsEntry 12 }

callSignalStatsReleaseCompleteMsgsIn OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      ~~current~~deprecated
    DESCRIPTION
        "The number of release complete messages received by this
         entity."
    ::= { callSignalStatsEntry 13 }
callSignalStatsReleaseCompleteMsgsOut OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      ~~current~~deprecated
    DESCRIPTION
        "The number of release complete messages sent by this
         entity."
    ::= { callSignalStatsEntry 14 }
callSignalStatsStatusMsgsIn OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      ~~current~~deprecated
    DESCRIPTION
        "The number of status messages received by this entity."
```

```
    ::= { callSignalStatsEntry 15 }
callSignalStatsStatusMsgsOut OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      ~~current~~deprecated
    DESCRIPTION
        "The number of status messages sent by this entity."
    ::= { callSignalStatsEntry 16 }
callSignalStatsStatusInquiryMsgsIn OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      ~~current~~deprecated
    DESCRIPTION
        "The number of status inquiry messages received by this
         entity."
    ::= { callSignalStatsEntry 17 }

callSignalStatsStatusInquiryMsgsOut OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      ~~current~~deprecated
    DESCRIPTION
        "The number of status inquiry messages sent by this
         entity."
    ::= { callSignalStatsEntry 18 }
callSignalStatsFacilityMsgsIn OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      ~~current~~deprecated
    DESCRIPTION
        "The number of connect messages received by this entity."
    ::= { callSignalStatsEntry 19 }
callSignalStatsFacilityMsgsOut OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      ~~current~~deprecated
    DESCRIPTION
        "The number of connect messages sent by this entity."
    ::= { callSignalStatsEntry 20 }
callSignalStatsInfoMsgsIn OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      ~~current~~deprecated
    DESCRIPTION
        "The number of info messages received by this entity."
    ::= { callSignalStatsEntry 21 }
callSignalStatsInfoMsgsOut OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      ~~current~~deprecated
    DESCRIPTION
        "The number of info messages sent by this entity."
    ::= { callSignalStatsEntry 22 }

callSignalStatsNotifyMsgsIn OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      ~~current~~deprecated
    DESCRIPTION
        "The number of notify messages received by this entity."
    ::= { callSignalStatsEntry 23 }
```

```
callSignalStatsNotifyMsgsOut OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      ~~current~~deprecated
    DESCRIPTION
        "The number of notify messages sent by this entity."
    ::= { callSignalStatsEntry 24 }

callSignalStatsAverageCallDuration OBJECT-TYPE
         SYNTAX      Integer32
         MAX-ACCESS  read-only
         STATUS      current
         DESCRIPTION
             "The average duration of the call in minutes since
              system boot time. "
         ::= { callSignalStatsEntry 25 }

callSignalStatsCallConnections OBJECT-TYPE
               SYNTAX  Counter32
               MAX-ACCESS  read-only
               STATUS  current
               DESCRIPTION
                  "The number of successful connections ."
                   ::= { callSignalStatsEntry 26 }

callSignalStatsAlertingMsgs OBJECT-TYPE
               SYNTAX  Counter32
               MAX-ACCESS  read-only
               STATUS  current
               DESCRIPTION
                  "The number of alerting messages."
                   ::= { callSignalStatsEntry 27 }

callSignalStatsCallProceedings OBJECT-TYPE
               SYNTAX  Counter32
               MAX-ACCESS  read-only
               STATUS  current
               DESCRIPTION
                  "The number of call proceeding messages."
                   ::= { callSignalStatsEntry 28 }

callSignalStatsSetupMsgs OBJECT-TYPE
               SYNTAX  Counter32
               MAX-ACCESS  read-only
               STATUS  current
               DESCRIPTION
                  "The number of setup messages."
                   ::= { callSignalStatsEntry 29 }

callSignalStatsSetupAckMsgs OBJECT-TYPE
               SYNTAX  Counter32
               MAX-ACCESS  read-only
               STATUS  current
               DESCRIPTION
                  "The number of setupAck messages."
                   ::= { callSignalStatsEntry 30 }

callSignalStatsProgressMsgs OBJECT-TYPE
               SYNTAX  Counter32
               MAX-ACCESS  read-only
               STATUS  current
```

```
                DESCRIPTION
                    "The number of progress messages."
                     ::= { callSignalStatsEntry 31 }


callSignalStatsReleaseCompleteMsgs OBJECT-TYPE
                SYNTAX  Counter32
                MAX-ACCESS  read-only
                STATUS  current
                DESCRIPTION
                    "The number of release complete messages."
                     ::= { callSignalStatsEntry 32 }

callSignalStatsStatusMsgs OBJECT-TYPE
                SYNTAX  Counter32
                MAX-ACCESS  read-only
                STATUS  current
                DESCRIPTION
                    "The number of status messages."
                     ::= { callSignalStatsEntry 33 }


callSignalStatsStatusInquiryMsgs OBJECT-TYPE
                SYNTAX  Counter32
                MAX-ACCESS  read-only
                STATUS  current
                DESCRIPTION
                    "The number of status inquiry messages."
                     ::= { callSignalStatsEntry 34 }

callSignalStatsFacilityMsgs OBJECT-TYPE
                SYNTAX  Counter32
                MAX-ACCESS  read-only
                STATUS  current
                DESCRIPTION
                    "The number of connect messages."
                     ::= { callSignalStatsEntry 35 }

callSignalStatsInfoMsgs OBJECT-TYPE
                SYNTAX  Counter32
                MAX-ACCESS  read-only
                STATUS  current
                DESCRIPTION
                    "The number of info messages."
                     ::= { callSignalStatsEntry 36 }

callSignalStatsNotifyMsgs OBJECT-TYPE
                SYNTAX  Counter32
                MAX-ACCESS  read-only
                STATUS  current
                DESCRIPTION
                    "The number of notify messages."
                     ::= { callSignalStatsEntry 37 }
```

[End Correction]

5.7.2 Corrections to H.341 Annex B-2 RAS-MIB

Description:	A few editorial errors have been identified in the RAS MIB in H.341. The **rasAdmissionCallIdentifier** field is inserted twice in the **RasAdmissionTableEntry** SEQUENCE. The ASN type of **rasRegistrationEndpointType** field in **RasRegistrationTableEntry** SEQUENCE should be changed to **MmH323EndpointType**.

[Begin Correction]

```
RAS-MIB DEFINITIONS ::= BEGIN
         IMPORTS
           Counter32, Integer32, OBJECT-TYPE,
           MODULE-IDENTITY, NOTIFICATION-TYPE
                 FROM SNMPv2-SMI
           TAddress, RowStatus, TruthValue, DateAndTime
                 FROM SNMPv2-TC
           MODULE-COMPLIANCE, OBJECT-GROUP
                 FROM SNMPv2-CONF
           ifIndex
                 FROM IF-MIB
           MmGatekeeperID, MmTAddressTag,MmEndpointID, MmGlobalIdentifier,
           MmAliasTag, MmAliasAddress, mmH323Root, MmH323EndpointType,
           MmH225Crv, MmCallType
                 FROM MULTI-MEDIA-MIB-TC;

...
         RasRegistrationTableEntry ::=
              SEQUENCE {
              rasRegistrationCallSignallingAddressTag
                    MmTAddressTag,
              rasRegistrationCallSignallingAddress
                    TAddress,
              rasRegistrationSrcRasAddressTag
                    MmTAddressTag,
              rasRegistrationSrcRasAddress
                    TAddress,
              rasRegistrationIsGatekeeper
                    TruthValue,
              rasRegistrationGatekeeperId
                    MmGatekeeperID,
              rasRegistrationEndpointId
                    MmEndpointID,
              rasRegistrationEncryption
                    TruthValue,
              rasRegistrationWillSupplyUUIE
                    TruthValue,
              rasRegistrationIntegrityCheckValue
                    TruthValue,
              rasRegistrationTableNumberOfAliases
                    Integer32,
              rasRegistrationTableRowStatus
                    RowStatus,
              rasRegistrationEndpointType
                    MmH323EndpointTypeInteger32,
              rasRegistrationPregrantedARQ
                    TruthValue,
              rasRegistrationIsregisteredByRRQ
                    TruthValue
```

```
            }
...
             rasRegistrationEndpointType OBJECT-TYPE
                  SYNTAX MmH323EndpointTypeInteger32
                  MAX-ACCESS read-only
                  STATUS current
                  DESCRIPTION
                    "Terminal type represents the type of H.323 terminal:
...
              RasAdmissionTableEntry ::=
                  SEQUENCE {
                  rasAdmissionSrcCallSignallingAddressTag
                        MmTAddressTag,
                  rasAdmissionSrcCallSignallingAddress
                        TAddress,
                  rasAdmissionDestCallSignallingAddressTag
                        MmTAddressTag,
                  rasAdmissionDestCallSignallingAddress
                        TAddress,
                  rasAdmissionCallIdentifier
                        MmGlobalIdentifier,
                  rasAdmissionConferenceId
                        MmGlobalIdentifier,
                  rasAdmissionRasAddressTag
                        MmTAddressTag,
                  rasAdmissionRasAddress
                        TAddress,
                  rasAdmissionCRV
                        MmH225Crv,
                  rasAdmissionIsGatekeeper
                         TruthValue,
                  rasAdmissionSrcAliasAddressTag
                        MmAliasTag,
                  rasAdmissionSrcAliasAddress
                        MmAliasAddress,
                  rasAdmissionDestAliasAddressTag
                        MmAliasTag,
                  rasAdmissionDestAliasAddress
                        MmAliasAddress,
                  rasAdmissionAnswerCallIndicator
                        INTEGER,
                  rasAdmissionTime
                        DateAndTime,
                  rasAdmissionCallIdentifier
                        MmGlobalIdentifier,
                  rasAdmissionEndpointId
                        MmEndpointID,
                  rasAdmissionBandwidth
                        Integer32,
                  rasAdmissionIRRFrequency
                        Integer32,
                  rasAdmissionCallType
                         MmCallType,
                  rasAdmissionCallModel
                        INTEGER,
                  rasAdmissionSrcHandlesBandwidth
                        TruthValue,
                  rasAdmissionDestHandlesBandwidth
                        TruthValue,
```

```
                rasAdmissionSecurity
                      TruthValue,
                rasAdmissionSrcWillSupplyUUIE
                      TruthValue,
                rasAdmissionDestWillSupplyUUIE
                      TruthValue,
                rasAdmissionTableRowStatus
                      RowStatus
                }
```

[End Correction]

5.7.3 Support for Expanded Country Code Values in T.35 in H.341 Annex B-3

Description:	T.35 (1999) expanded the available country codes from one octet to two octets. In order to support the expanded country codes going forward, it is recommended that implementers make the following changes to these definitions in H.341 Annex B-3 H323TERMINAL-MIB.

[Begin Correction]

```
 h323TermSystemt35CountryCode OBJECT-TYPE
             SYNTAX INTEGER (0..255)
             MAX-ACCESS read-only
             STATUS current
             DESCRIPTION
         "Country code, per T.35 Annex A."
 ::= { h323TermSystemEntry 5 }
 h323TermSystemt35CountryCodeExtention OBJECT-TYPE
             SYNTAX INTEGER (0..255)
             MAX-ACCESS read-only
             STATUS current
             DESCRIPTION
"Assigned nationally, unless the country code
is 255, in which case this value shall contain
the country code found in T.35 Annex B."
::= { h323TermSystemEntry 6 }
```

[End Correction]

5.8 Technical and Editorial Corrections to ITU-T Recommendation H.283 (1999)

5.8.1 Support for Expanded Country Code Values in T.35

Description:	T.35 (1999) expanded the available country codes from one octet to two octets. In order to support the expanded country codes going forward, it is recommended that implementers take note of the following usage guidelines for fields in H.283.

[Begin Correction]

```
                                    ...
H221NonStandard ::= SEQUENCE
{
    t35CountryCode     INTEGER(0..255),    -- country, as per T.35 Annex A
```

```
    t35Extension        INTEGER(0..255),    -- assigned nationally, unless the
                                                -- t35CountryCode is binary 1111
    1111,
                                             -- in which case this field shall
                                                -- contain the country code found
                                                -- in T.35 Annex B
    manufacturerCode    INTEGER(0..65535)   -- assigned nationally
}
```

...

[End Correction]

5.9 Technical and Editorial Corrections to ITU-T Recommendation H.460 Series

5.9.1 Technical and Editorial Corrections to H.460.1 (2002)

5.9.1.1 Encoding rules

Description:	Currently, H.460.1 requires each individual Recommendation to specify the encoding that it will use for parameters in the raw format. It would be better to specify the default encoding in H.460.1 and let individual Recommendations change it only if needed. The textual changes are shown below.

[Begin Correction]

7.2 Encoded in Raw Method

…

If the feature is defined using ASN.1, then it is recommended that the basic aligned variant of the PER encoding rules be used. However, irrespective of this, ~~the~~ encoding rules that are used, if different from the above, shall be explicitly stated in the specification of the feature.

[End Correction]

5.9.2 Technical and Editorial Corrections to H.460.2 (2001)

5.9.2.1 Typographical Error in Section 4.1

Description:	A typographical error has been discovered in that the parameter **qorPortedNumber** in the ASN.1 is referred to as **qorPortedAddress** in Section 4.1 of H.460.2 (2001). The text below outlines the necessary change.

[Begin Correction]

4.1 Messages and Signaling

…

5) When a Gatekeeper receives an ARQ or LRQ and determines that the destination number is ported out of the network and it may wish to invoke number portability Query on Release (QoR) procedures (as specified in Annex C/Q.769.1). In such cases, the Gatekeeper must respond with ARJ or LRJ that contains a reject reason of **genericDataReason**. The Gatekeeper should include the **genericData** of the ARJ/LRJ that contains the **NumberPortabilityGenericData** with the **numberPortabilityRejectReason**. The **numberPortabilityRejectReason** now will have a value of **qorPortedNumber~~Address~~** (=1). This maps to the ISUP release cause value = #14 (QoR: ported number) as specified in Addendum 1/Q.850.

[End Correction]

5.9.2.2 Cardinality of Number

Description:	A typographical error has been discovered in that the parameter **qorPortedNumber** in the ASN.1 is referred to as **qorPortedAddress** in Section 4.1 of H.460.2 (2001). The text below outlines the necessary change.

[Begin Correction]

5 H.225.0 Generic Data Usage

…

Generic Extensibility Type	**Fields**	**Field name**	**Value**
EnumeratedParameter			
GenericIdentifier	**id**	**standard**	1
Contents	**content**	**raw**	ASN.1 PER encoding of the **NumberPortabilityInfo**
Parameter Cardinality			Once and Only Once

[End Correction]

5.9.3 Technical and Editorial Corrections to H.460.6 (2002)

5.9.3.1 Close All Channels

Description:	The intent of the Close All Media Channels request described in section 4.1.2 is to close all open media channels and cancel all available sessions, as described in section 4.5. To this end, text in sections 4.1.2 and 4.5.2 should be changed as follows.

[Begin Correction]

4.1.2 Close All Channels

This parameter may be used by a party to request that the receiving endpoint close all open media channels and cancel all available sessions. Support for this parameter is optional, and shall be negotiated during EFC feature negotiation.

…

4.5.2 Requesting Close-All-Channels

An endpoint or a third party may request that the other endpoint close all open media channels and cancel all available sessions by sending a **genericData** element with the EFC featureID and parameter 2 present in any convenient call signalling message (e.g., FACILITY). The receiving endpoint is expected to silently close all open channels without any response (e.g., without issuing any **Null-OLC**s.)

[End Correction]

5.9.3.2 Signaling of EFC Support in supportedFeatures

Description:	It is held that signalling of EFC in **supportedFeatures** by the originating party is unnecessary. The text in section 4.2 should be corrected as below.

[Begin Correction]

4.2 Invocation of Extended Fast Start

An originating party shall indicate its desire to use EFC when it issues a SETUP message. The SETUP shall contain a request for EFC support in the **desiredFeatures** element, or a requirement for EFC support in the **neededFeatures** element. ~~The **supportedFeatures** element shall indicate support for EFC as well.~~ The EFC feature is symmetric, hence requestor support for the feature may be inferred from a request for EFC, and the **supportedFeatures** element need not be included to indicate support for EFC. In addition, the SETUP message shall include a **genericData** element specifying EFC Proposal (parameter 1) and a **fastStart** element containing one or more proposals. That is, EFC procedures shall include the standard Fast Connect procedures.

[End Correction]

5.9.3.3 Prevention of Race Condition in Master/Slave Determination

Description:	There is a possible race condition that may occur, depending on the order in which an endpoint processes **fastStart** elements versus tunnelled H.245 master/slave negotiation messages embedded in the same H.225.0 message. Thus, it is suggested that the following paragraph be added to the end of section 4.2.1.

[Begin Correction]

4.2.1 Master/Slave Determination

Parties supporting Extended Fast Connect should use the H.245 tunnel to carry out master/slave negotiation. For the initial Fast Connect exchange, the caller (sender of the SETUP with proposals) shall be considered the slave, and the called party (acceptor of proposals) shall act as the master. Although this convention will suffice for simple A-to-B calls, it can lead to complications in more complex call scenarios.

Different implementations may process **fastStart** elements and tunnelled H.245 messages in different orders. EFC proposals or acceptances shall not be included in any H.225.0 message that carries an H.245 **MasterSlaveDeterminationAck** message that conveys a change in master/slave status. Doing so could lead to temporary confusion about which party is master and how to respond to the EFC elements.

[End Correction]

5.9.3.4 Remote Endpoint Type and Version after Re-routing

Description:	An endpoint may not be aware of the H.323 protocol version number supported by the remote endpoint, especially if the call gets re-routed one or more times. In some cases it might be helpful for the endpoint to have this information. The following additions should be made to H.460.6 document.

[Begin Correction]

5.5 EFC Third-party Pause and Rerouting

EFC supports third-party pause and rerouting, as described in H.323 Annex F for SETs, when used by a routing gatekeeper. The third party (the gatekeeper in the example in Figure 5) may idle the caller's transmit and/or receive channels via **Null-OLC**s, then supply the caller's proposal **fastStart** to a new party (e.g., in a SETUP). The acceptance **fastStart** will appear to the caller as a redirection or reconfiguration, asillustrated in Figure 5.

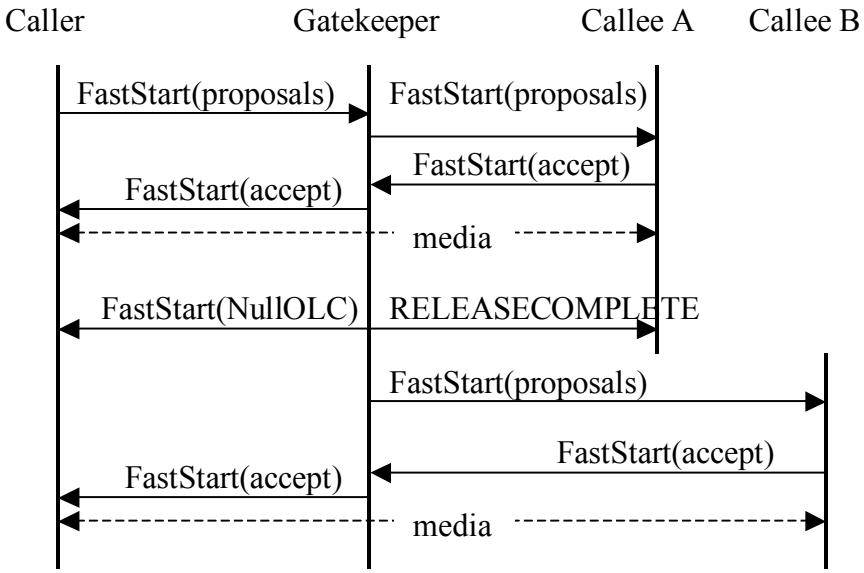


Figure 5/H.460.6 - Third-Party Redirection

In the above diagram, the Gatekeeper, or the entity that re-routes a call should send a Facility message containing the **destinationInfo** field upon completion of the re-routing to the entity that gets re-routed, i.e., Caller. An endpoint should examine this message for the H.225.0 version information at any point that a Facility message is received containing this field.

After coming out of the "paused" state an endpoint should examine the version-id fields in TCS messages to determine the H.245 version supported by the remote endpoint.

In addition, an endpoint interested in knowing the version of the remote endpoint should send a Status Inquiry message and wait for the receipt of the Status message to determine the version of the H.225.0 in use when it exits the paused state when the above Facility message is not received within a reasonable amount of time. The length of this time is left to the implementation.

[End Correction]

5.9.3.5 Termination of Extended Fast Connect

Description:	It is held that extended fast connect shall not be terminated when an H.245 address is present in a message returned by the called party. Instead, EFC shall be terminated when a connection is established to the H.245 address. To this end, text in section 4.2 should be changed as follows.

[Begin Correction]

4.2 Invocation of Extended Fast Connect

…

Extended Fast Connect ~~Start~~ requires that H.245, if used, shall be tunnelled in the H.225.0 signalling channel. If a calling party offers Extended Fast Connect ~~Start~~ support in SETUP, and the called party returns a message including **h245Address** before returning an EFC response, ~~(thereby requesting a separate H.245 connection),~~ the calling party shall presume that EFC is not supported and may proceed with establishment of the requested connection. Including **h245address** in messages does not by itself terminate EFC, but establishing a connection to an H.245 address will cause termination of EFC.

[End Correction]

5.9.3.6 Clarification on simultaneous use of EFC acceptance and acceptance fastStart

Description:	EFC acceptance and acceptance **fastStart** can be simultaneously included in any message up to and including Connect. Having both the fields in the same message can cause undue processing on the recipient and must be avoided when it is possible. To this end, text in section 4.2 should be changed as follows.

[Begin Correction]

4.2 Invocation of Extended Fast Connect

…

Note that the EFC acceptance and acceptance **fastStart** can be returned in any message up to and including the CONNECT message, but the identical acceptance should not be repeated in any subsequent message.

[End Correction]

5.9.3.7 Clarification on orientation of logical channels

Description:	Orientation of logical channels proposed using EFC are not related to the direction of the call. The following paragraph should be added to Section 4.3 to clarify this.

[Begin Correction]

4.3 Opening New Sessions

Opening new media sessions proceeds just like standard Fast Connect, except that either party may invoke EFC at any time to propose new media sessions by sending a message (e.g., FACILITY) containing a proposal **fastStart** element with one or more **OLC**s for one or more **sessionID**s, along with a **genericData** element indicating "EFC Proposal". As for standard Fast Connect, multiple **OLC**s with the same **sessionID** are considered to be alternative proposals for a single media stream. The other party may reply with a **fastStart** element containing **OLC**s for the accepted channels and sessions. A slave party will supply a non-zero **sessionID** for any media channels it proposes. Parties may use EFC to propose and open any number of sessions. Session IDs may take any valid value and need not be limited to the "well-known" values of 1, 2, or 3.

For each logical channel, the EFC proposal establishes the orientation of the forward and reverse logical channels: the forward logical channel carries media from the proposer to the acceptor, and the reverse logical channel carries media from the acceptor to the proposer; the order is not determined from the direction of the call as a whole.

As in standard Fast Connect, once a proposed alternative is selected by another party, the issuing endpoint may suspend any reception of media on the other alternatives. Nevertheless, it shall be prepared for the other party to replace the initially-selected alternative with another (see section 4.7, below).

…

[End Correction]

5.9.4 Technical and Editorial Corrections to H.460.7 (2002)

5.9.4.1 Compound Type Parameter Usage

Description:	The contents of the compound parameter in Section 6.4, Table 9 are not well defined. The text below clarifies its usage.

[Begin Correction]

6.4 Digit map string parameters

…

Table 9/H.460.7 – Type of Number Associated Digit Maps

Parameter name:	**ToN Associated Digit Map**
Parameter description:	This compound type conveys Digit Map associated with a particular Type of Number
Parameter identifier type:	Standard
Parameter identifier value:	5
Parameter type:	Compound
Parameter cardinality:	Zero or more

Within the **compound** type defined in Table 9, the parameters defined in Table 2 and Table 11 shall be included to convey one or more Digit Map strings for a particular Type of Number:

Table 10/H.460.7 – Type of Number Parameter

Parameter name:	**Type of Number** (ToN)
Parameter description:	This parameter indicates the type of number
Parameter identifier type:	Standard
Parameter identifier value:	1
Parameter type:	Number8
Parameter valid values:	1 International number 2 National number 3 Network specific number 4 Subscriber number 6 Abbreviated number
Parameter cardinality:	Once

The Digit Map strings comprising the Digit Map associated with a Type of Number are conveyed as additional parameters within the **compound** type of the Type of Number Associated Digit Maps parameter shown in Table ~~11~~9. This is shown in Table 11.

Table 11/H.460.7 - Digit Map strings for ToN Parameter

Parameter name:	**Digit Map Strings for ToN**
Parameter description:	This parameter contains a single Digit Map string
Parameter identifier type:	Standard
Parameter identifier value:	2
Parameter type:	Text
Parameter cardinality:	One or more

The syntax of the **text** field, which holds a single Digit Map string, is described in section 10.

The order of the Digit Map strings in the **parameters** field has no significance.

[End Correction]

5.9.4.2 Duplicate Parameters

Description:	Section 6.3 (Table 3) and Section 6.5 (Table 12) both define a parameter with ID 2. They can be used in the same place (in an RCF), so it is not possible to distinguish between them. The parameter identifier value should be changed to 2 as below.

[Begin Correction]

6.5 URL parameter

…

Table 12/H.460.7 - URL Parameter

Parameter name:	**Digit Map URL**
Parameter description:	This parameter contains a URL to Digit Map information accessible via HTTP
Parameter identifier type:	Standard
Parameter identifier value:	~~2~~6
Parameter type:	Alias
Parameter cardinality:	Zero or one

[End Correction]

5.9.6 Technical and Editorial Corrections to H.460.18 (2005)

5.9.6.1 Editorial corrections to clause 4 – Abbreviations

Description:	Some abbreviations in Clause 4 need corrections. The text below specifies these corrections.

[Begin Correction]

This Recommendation uses the following abbreviations:

ACF Admission Confirm~~ation~~ (H.225.0)

ARQ ~~Automatic Repeat~~Admission Request (H.225.0)

…

[End Correction]

5.9.6.2 Editorial corrections to clause 8.1 – Traversal server mode selection

Description:	The text "supported features" is changed to reflect how it appears elsewhere in the Recommendation. The text of clause 8.1 is corrected as follows.

[Begin Correction]

If the TS has prior knowledge that there is no NAT/FW between itself and the endpoint, it may elect not to use the procedures described in this Recommendation. If NAT/FW traversal is not required, the TS may omit **Signalling Traversal** from the **supportedFeatures**~~supported features~~ field of the RCF. Signalling then proceeds without the procedures described in this Recommendation.

[End Correction]

5.9.6.3 Editorial corrections to clause 8.2 – Registration when H.460.18 mode selected by traversal server

Description:	Typographical errors appear in the title and the text. The text of clause 8.2 is corrected as follows.

[Begin Correction]

8.2 Registration when H.460.18 mode selected by traversal server

…

If the TS accepts a gatekeeper discovery or registration, it shall send a GCF or RCF with **Signalling Traversal** in the **supportedFeatures** field. The TS shall set the **timeToLive** in the RCF to a value that is short enough to prevent intermediate NAT/FW devices from blocking connectivity. This value shall be determined as described in clause 14.

…

[End Correction]

5.9.6.4 Editorial corrections to Figure 4 – Indicative outgoing call message sequence

Description:	Figure 4 incorrectly displays H.225 instead of H.225.0. The text in the figure is corrected as follows.

[Begin Correction]

Figure 4/H.460.18 Indicative Outgoing Call Message Sequence

[End Correction]

5.9.6.5 Editorial corrections to clause 10 – Incoming call procedure

Description:	The text below corrects a typographical error in Clause 10.

[Begin Correction]

EP_A is located on the internal network, EP_B is on the external network. EP_B is H.323 conformant and is outside the scope of this Recommendation.

1) To establish a call to EP_A in the internal network (for example, in response to an H.225.0 call setup from ~~a~~ EP_B), the TS shall send an H.225.0 SCI RAS message to EP_A. The **genericData** field of the SCI shall contain an **IncomingCallIndication** as defined in Table 2.

…

[End Correction]

5.9.6.6 Editorial corrections to Figure 5 – Indicative incoming call message sequence

Figure 5 incorrectly displays H.225 instead of H.225.0 and contains some typographical errors. The text in the figure is corrected as follows:

[Begin Correction]

Figure 5/H.460.18 Indicative Incoming Call Message Sequence

TS

[End Correction]

5.9.7 Technical and Editorial Corrections to H.460.19 (2005)

5.9.7.1 Editorial corrections to Scope

Description:	The text would be modified to (i) clarify that H.460.19 addresses the following types of media streams. o RTP stream o RTP stream encrypted using H.235 o SRTP stream (ii) clarify minor editorial issues.

[Begin Correction]

3 Definition

…

This Recommendation addresses NAT/FW traversal for the following types of media streams.

1) RTP
2) RTP encrypted using H.235
3) SRTP~~NAT/FW traversal for RTP; RTP encrypted using H.235 and SRTP media streams only.~~

NAT/FW traversal for media transported by other protocols is for further study.

It also defines a mechanism to use the same transport address for several media channels, which permits reduction of the number of "pinholes" opened in the NAT/FW device and reduces the number of Media Channel and Media Control Channel transport addresses used by H.323 entities.

[End Correction]

5.9.7.2 Editorial corrections to Section 3/H.460.19

Description:	(i) Minor editorial clarifications. (ii) There is a case in Fast Connect that an "OLC response" message is carried earlier in time than the corresponding "OLC request" message. A note is added to clarify what constitutes and OLC request and an OLC response as used in this specification. (iii) Added definition for ‘demilitarized zone’.

[Begin Correction]

3 Definitions

…

3.3 demilitarized zone: a network that sits between an organization's internal network and an external network, usually the Internet. Connections from the internal and the external network to the DMZ are permitted, whereas connections from the DMZ are only permitted to the external network

…

3.45 external network: A network connected to the firewall through ~~the firewalls~~its public interface~~.~~; ~~T~~typically, but not limited to, the Internet.

3.~~5~~6 H.460.19 entity: A client or server.

3.~~6~~7 internal network: A network connected to the NAT/FW through the NAT/FW's private interface.

3.~~7~~8 media channel: An RTP or an SRTP channel.

3.~~8~~9 media control channel: An RTCP or SRTCP channel.

3.~~9~~10 multiplexed media mode: A mechanism which enables managing multiple RTP/RTCP or SRTP/SRTCP sessions on a single pair of transport addresses as described in 7.2. Receivers choose whether or not to multiplex.

3.~~10~~11 OLC request: Any one of:

- the **openLogicalChannel** message;
- the **openLogicalChannel** Fast Connect proposal message for transmission from caller to callee;
- the **openLogicalChannel** Fast Connect proposal accept message for transmission from callee to caller.

3.~~11~~12 OLC response: Any one of:

- the **openLogicalChannelAck** message;
- the **openLogicalChannel** Fast Connect proposal accept message for transmission from caller to callee;
- the **openLogicalChannel** Fast Connect proposal message for transmission from callee to caller.

NOTE: An OLC request message is the message that contains information relevant to the OLC for the media stream that is received by the addressee of this OLC request message, while an OLC response message is the message that contains information relevant to the OLC for the media stream that is sent by the addressee of this OLC response message.

3.~~12~~13 peer: An H.460.19 entity with which a particular H.460.19 entity is communicating.

3.~~13~~14 pinhole: A temporary binding of an internal and an external transport address in the NAT/FW which allows the bidirectional passage of packets between those addresses.

3.~~14~~15 server: An H.323 entity compliant with H.460.19 specifications and performing H.460.19 server functionality.

3.~~15~~16 transport address: IP address and UDP/TCP port number.

[End Correction]

5.9.7.3 Editorial corrections to Section 4/H.460.19

Description:	Minor editorial clarifications

[Begin Correction]

4 Abbreviations

…

RTCP Real-t~~T~~ime Transport Control Protocol (according to RFC 3550)

RTP Real-~~T~~time Transport Protocol (according to RFC 3550)

SRTCP Secure Real-~~T~~time Transport Control Protocol (according to RFC 3711)

SRTP Secure Real-t~~T~~ime Transport Protocol (according to RFC 3711)

[End Correction]

5.9.7.4 Editorial corrections to Section 6/H.460.19

Description:	Minor editorial clarifications

[Begin Correction]

6 Architecture

…

Media Channel traffic is unidirectional. To permit Media Channel packets toward the internal network, this Recommendation defines a ~~Keep-Alive~~keep-alive ~~C~~channel. The client sends ~~Keep-Alive~~keep-alive media packets to the **keepAliveChannel** transport address provided by the server.

In the NAT/FW traversal procedure, the H.460.19 server sends Media Channel and Media Control Channel packets to the H.460.19 client to the address from which Keep Alive Channel and Media Control Channel packets were received by the H.460.19 server, instead of to the addresses specified in the **H2250LogicalChannelParameters.mediaChannel** and **H2250LogicalChannelParameters.mediaControlChannel** H.245 structures as in normal H.323 operation. The NAT/FW traversal procedure also requires usage of a keep-alive mechanism. The goal of the keep-alive mechanism is to ensure that there are no extended periods of "network silence" between the communicating Transport Addresses, which might result in closure of pinholes by the NAT/FW. ~~The i~~Implementation of the ~~K~~keep-~~A~~alive mechanism is mandatory in client to server direction.

…

In Figure 3, Organization A on the left has a mix of H.460.19-enabled endpoints, and of non-H.460.19 H.323 endpoints making use of an H.460.19 proxy. Organization A has an H.460.19 server on their demilitarized zone (shown as part of the external network in the figure), which provides media traversal and access to the external network for Organization A endpoints.

…

6.1 General requirements

…

Support of ~~transmission of the~~ multiplexed media mode transmission defined in 7.3.2 is mandatory for H.460.19 clients and optional for H.460.19 servers.

Support of ~~reception of the~~ multiplexed media mode reception defined in 7.3.2 is optional for both H.460.19 clients and H.460.19 servers.

[End Correction]

5.9.7.5 Editorial corrections to Section 7/H.460.19

Description:	(i) Minor editorial corrections (ii) Keep-alives have to be sent at an interval less than the one specified by the server, but the text indicates the opposite. This is corrected. (iii) A note advising against the use of very short keep-alive intervals is added.

[Begin Correction]

7.1.1 Capabilities signalling

…

The capability to transmit in multiplexed media mode shall be signalled by the servers by including the **supportTransmitMultiplexedMedia** parameter, defined in 7.4.2, in the same **supportedFeatures** field.

…

The capability to receive in multiplexed media mode (to de-multiplex) shall be indicated by the presence of the **multiplexID** field in the Traversal Parameters in the OLC Request and OLC Response messages as defined in 7.4.5.

…

7.1.2 Logical channel signalling

…

In all cases, the H.460.19 client shall send ~~K~~keep-~~A~~alive packets as defined in 7.3.1.1.

…

Table 2/H.460.19 – Transport addresses for channels between H.460.19 client and server

Channel	Source	Source transport address	Dest.	Destination transport address
Media Channel	client	Any port on H.460.19 client.	server	**mediaChannel** destination address on H.460.19 server in server's OLC Response message.
Media Channel	server	**keepAliveChannel** destination address on H.460.19 server in server's OLC Request message.	client	Apparent k~~K~~keep-~~A~~alive source address on H.460.19 client (media sent only after receipt of ~~K~~keep-~~A~~alive from H.460.19 client).
Keep-~~A~~alives	client	H.460.19 client's desired Media Channel destination port~~ on H.460.19 client~~.	server	**keepAliveChannel** destination address on H.460.19 server in server's OLC Request message.
Media Control Channel	client	H.460.19 client's desired Media Control Channel destination port~~. on the client.~~ The **mediaControlChannel** destination address in all OLC Request and OLC Response messages sent by a given client for a given call and value of **sessionID** shall contain this same transport address. NOTE – The H.460.19 server ignores this **mediaControlChannel** value.	server	**mediaControlChannel** destination address on H.460.19 server in server's OLC Request or OLC Response message – whichever was received more recently for the given call and value of **sessionID**.
Media Control Channel	server	**mediaControlChannel** destination address on H.460.19 server in H.460.19 server's OLC Request or OLC Response message – whichever was transmitted more recently for the given call and value of **sessionID**.	client	Apparent Media Control Channel source address on H.460.19 client (media control ~~channel~~ packets are sent only after receipt of media control packets from H.460.19 client).

…

7.1.2.2 Establishment of LCs from H.460.19 server to H.460.19 client

…

The H.460.19 client shall transmit Media Control Channel and ~~K~~keep-~~A~~alive packets to the transport addresses indicated in the **mediaControlChannel** and **keepAliveChannel** fields, respectively, of the H.460.19 server's OLC Request message.

The H.460.19 server shall wait for receipt of at least one ~~K~~keep-~~A~~alive packet from the H.460.19 client for the LC, and then send Media Channel packets for the LC to the H.460.19 client, with a destination transport address equal to the apparent source transport address of the k~~K~~eep-~~A~~alive packet received from the H.460.19 client.

…

7.1.2.3 Overlapping establishment of LCs between H.460.19 client and H.460.19 server

…

The H.460.19 client shall transmit Media Channel, Media Control Channel, and k~~K~~keep-~~A~~alive packets to the transport addresses indicated in the **mediaChannel**, **mediaControlChannel**, and **keepAliveChannel** fields, respectively, of the H.460.19 server's OLC Request or OLC Response message, whichever was received more recently for the same call and using the same value of **sessionID**.

The H.460.19 server shall wait for receipt of at least one ~~K~~keep-~~A~~alive packet from the H.460.19 client for the LC, and then send Media Channel packets for the LC to the H.460.19 client, with a destination transport address equal to the apparent source transport address of the ~~K~~keep-~~A~~alive packet received from the H.460.19 client.

…

7.1.2.4 Establishment of LCs from H.460.19 client to H.460.19 client

…

The H.460.19 client shall not transmit ~~K~~keep-a~~A~~live packets when the peer is also H.460.19 client.

…

7.1.2.5 Establishment of LCs from H.460.19 server to H.460.19 server

…

The H.460.19 server shall not transmit k~~K~~eep-~~A~~alive packets when the peer is also H.460.19 server.

…

7.2.1 Requesting the multiplexed media mode

…

An H.460.19 entity "A" (either an H.460.19 server or an H.460.19 client) may initiate multiplexing of the Media, Media Control and k~~K~~eep-~~A~~alive channels sent from an H.460.19 client to entity A.

An H.460.19 entity "A" (either an H.460.19 server or an H.460.19 client) may initiate multiplexing of the Media, Media Control and ~~K~~keep-~~A~~alive c~~C~~hannels sent from an H.460.19 server to entity A, only if the H.460.19 server supports multiplexing as indicated by **supportTransmitMultiplexedMedia** parameter of the server's feature identifier as defined in 7.4.2.

…

NOTE 1 – A **multiplexID** field may be included in an OLC Request message in order to request multiplexing of the Media Control Channel and ~~K~~keep-~~A~~alive channel in the direction toward the requesting entity.

…

NOTE 2 – The pair of LCs for the given call and value of **sessionID** establishes a Media Channel and a Media Control Channel in the direction toward the client and a ~~K~~keep-~~A~~alive ~~C~~channel and a Media Control Channel in the direction toward the server. The packets, of these channels, sent towards either of the entities include the **multiplexID** provided by the entity.

If the **multiplexID** field was present in the message, the peer H.460.19 entity shall transmit Media Channel, Media Control Channel and ~~K~~keep-~~A~~alive packets for the LCs in the multiplexed mode, identifying each packet with the **multiplexID** value as defined in 7.3.2.

Entity A shall receive Media Channel, Media Control Channel and ~~K~~keep-~~A~~alive packets for the LCs in the multiplexed media mode, and use the received value of the **multiplexID** in each packet as defined in 7.3.2.

…

If an OLC Request message contains **multiplexID** and **keepAliveChannel** fields, then the **keepAliveChannel** field shall contain the destination transport address for the multiplexed ~~K~~keep-~~A~~alive packets.

…

7.3.1 NAT/FW traversal procedure

…

The Media Channel, Media Control Channel and k~~K~~eep-~~A~~alive ~~C~~channel are considered established for the purpose of the ~~K~~keep-~~A~~alive procedure when the H.460.19 entity receives the OLC Request or OLC Response message containing the transport address for each channel~~message containing the transport address for each respective channel.~~

…

7.3.1.1 NAT/FW traversal procedure – Clients

…

Upon establishment of each ~~K~~keep-~~A~~alive ~~C~~channel, the H.460.19 client shall transmit one Media Channel ~~K~~keep-~~A~~alive packet.

Upon establishment of each Media Control Channel, the H.460.19 client shall transmit one Media Control Channel ~~K~~keep-~~A~~alive packet.

For each established Media Control Channel and ~~K~~keep-~~A~~alive Channel, the H.460.19 client shall transmit a Media Channel and Media Control Channel keep-alive packet at intervals ~~of not~~ less than the value specified by the H.460.19 server in the **keepAliveInterval** field in the Traversal Parameters defined in 7.4.5, unless there is other traffic on the channel within the given interval.

A keep-alive interval in the range of 5 to 30 seconds should be used except in cases where it is known (for example, from the specifics of the network) that a longer interval will not result in the closure of pinholes.

NOTE: Keep-alive packets should not be sent in an unduly short interval compared to the **keepAliveInterval** value. This is to avoid waste of the network and processing resources.

7.3.1.1.1 RTP ~~K~~keep-~~A~~alive packet

The RTP keep-alive packet is an RTP packet with an empty payload field. The payload type value shall be equal to the value specified by the client in **keepAlivePayloadType** field in the Traversal Parameters defined in 7.4.5. The sequence number header field shall start from any arbitrary value and increment by one for each ~~K~~keep-~~A~~alive packet.

The SSRC and timestamp header fields may have arbitrary values.

7.3.1.1.2 RTCP ~~K~~keep-~~A~~alive packet

The RTCP keep-alive packet is an RTCP packet containing only an SR (sender report) specified in RFC 3550.~~an SR (sender report) only, as specified in RFC 3550.~~

7.3.1.1.3 SRTP ~~K~~keep-~~A~~alive packet

The SRTP keep-alive packet is the same as the RTP ~~K~~keep-~~A~~alive packet. In addition, the optional authentication tag (as defined by RFC 3711) should be added to the packet.

7.3.1.1.4 SRTCP ~~K~~keep-~~A~~alive packet

The SRTCP keep-alive packet is an SRTCP packet containing an SR (sender report) authenticated and optionally encrypted with the same parameters which are used for regular SRTCP packets in the same SRTP session.

7.3.1.2 NAT/FW Traversal procedure – Servers

All H.460.19 servers shall implement the NAT/FW Traversal procedure defined in this clause.

H.460.19 servers shall not forward any RTP or SRTP ~~K~~keep-~~A~~alive packet as defined in the previous clause to any H.323 endpoint which has not indicated support for the procedures of this Recommendation.

…

For each established Media Channel, the H.460.19 server entity shall wait for receipt of at least one ~~K~~keep-~~A~~alive media packet from the H.460.19 client and then send media packets destined for the H.460.19 client to the source transport address of the ~~K~~keep-~~A~~alive media packet received from the H.460.19 client.

…

7.3.3 Multiplexed media mode – SRTP/SRTCP

The multiplexed media mode for SRTP/SRTCP is identical to the RTP/RTCP procedure in the previous clause, with the 4-byte **multiplexID** inserted between the UDP header and the SRTP/SRTCP headers.

…

7.4.5.1 Traversal Parameters semantics

…

keepAliveChannel

Keep-a~~A~~live ~~C~~channel packets shall be sent to the transport address received in this field. This field is used only to specify the address for the ~~K~~keep-~~A~~alive packets which need to be sent in the opposite direction to that of the Media Channel. This field shall be specified only by H.460.19 servers in OLC Request messages.

keepAlivePayloadType

The ~~K~~keep-~~A~~alive ~~C~~channel packets shall have the payload type value equal to the value specified in this field by the sender of the ~~K~~keep-~~A~~alive packets. This field shall be specified in the OLC Response messages by the H.460.19 client communicating with H.460.19 server or with an entity whose H.460.19 type is still unknown.

keepAliveInterval

This value is signalled by an H.460.19 server and represents the maximum interval, in seconds, of the absence of the Media Channel or Media Control Channel packet traffic, ~~after~~ within which the corresponding ~~K~~keep-~~A~~alive packets shall be sent.

[End Correction]

6 Implementation Clarification

6.1 Token Usage in H.323 Systems

There has been some confusion on the usage of individual **CryptoH323Tokens** as passed in RAS messages. There are two main categories of **CryptoH323Tokens**; those used for H.235 procedures and those used in an application specific manner. The use of these tokens should be according to the following rules:

- All H.235 defined (e.g. **cryptoEPPwdHash**, **cryptoGKPwdHash**, **cryptoEPPwdEncr**, **cryptoGKPwdEncr**, **cryptoGKCert**, and **cryptoFastStart**). shall be utilized with the procedures and algorithms as described in H.235.
- Application specific or proprietary use of tokens shall utilize the **nestedcryptoToken** for their exchanges.
- Any **nestedcryptoToken** used should have a **tokenOID** (object identifier) which unambiguously identifies it.

6.2 H.235 Random Value Usage in H.323 Systems

The random value that is passed in xRQ/xCF sequence between endpoints and Gatekeepers may be updated by the Gatekeeper. As described in section 4.2 of H.235 this random value may be refreshed in any xCF message to be utilized by a subsequent xRQ messages from the endpoint. Due to the fact that RAS messages may be lost (including xCF/xRJ) the updated random value may also be lost. The recovery from this situation may be the reinitializing of the security context but is left to local implementation.

Implementations that require the use of multiple outstanding RAS requests will be limited by the updating of the random values used in any authentication. If the updating of this value occurs on every response to a request, parallel requests are not possible. One possible solution, is to have a logical "window" during which a random value remains constant. This issue is a local implementation matter.

6.3 Gateway Resource Availability Messages

The Resources Available Indication (RAI) is a notification from a gateway to a gatekeeper of its current call capacity for each H-series protocol and data rate for that protocol. The gatekeeper responds with a Resources Available Confirmation (RAC) upon receiving a RAI to acknowledge its reception. A Gatekeeper should ignore any RAI notifications (e.g. send no RAC) upon receiving a RAI which contains bogus information (i.e. a bad endpointIdentifier).

6.4 OpenLogicalChannel in fastStart

In the H.225.0 ASN.1, **fastStart** is defined as SEQUENCE OF OCTET STRING OPTIONAL. The text definition states "This uses the **OpenLogicalChannel** structure defined in H.245…" Each OCTET STRING in **fastStart** is to contain the **OpenLogicalChannel** structure, not an entire request message.

6.5 Clarification in Q.931 (1993)

Table 4-3/Q.931 (1993) (Information Element Identifier Coding) shows that the Progress Indicator IE identifier is 0x1e, but Figure 4-29/Q.931 (octet layout of Progress Indicator IE) shows the identifier as 0x1f. Note that the identifier should be 0x1e.

6.6 Graceful Closure of TCP Connections

When a TCP connection is closed, the graceful closure procedure documented in section 3.5 of RFC 793 should always be used.

6.7 Race Condition on Simultaneous Close of Channels

Section 8.5 of H.323 describes the procedures that an endpoint follows to terminate a call. It should be noted that as prescribed in Step 6, both endpoints shall issue a Release Complete simultaneously. Endpoints should be prepared for this potential race condition.

6.8 Acceptance of Fast Connect

When an endpoint accepts the Fast Connect procedure, it may select from the proposed channels as specified in section 8.1.7.1/H.323. The Recommendation clearly specifies what fields shall be modified by the endpoint to accept both the forward and the reverse channels. An endpoint shall not modify any fields other than those specified in 8.1.7.1/H.323 when returning the proposed channels.

Newer versions of H.245 may introduce new fields into the **OpenLogicalChannel** sequence or one of the structures contained therein, as well as new procedures. An older endpoint is obviously not required to decode such new fields or to return such new fields when accepting any proposal. Implementers should consider the consequences of transmitting a newer H.245 OLC to an older endpoint. For the purposes of Fast Connect, the calling endpoint shall assume that the called endpoint's version of H.245 is the minimum version of H.245 necessary to be complaint with an H.323 device that advertises the version of H.225.0 transmitted in the messages from the called endpoint (refer to the "Summary" section of H.323).

6.9 Semantic Differences between Lightweight RRQs and IRQ/IRR Messages

The lightweight RRQ and the IRR message serve two different functions with an H.323 system. While both are a means of allowing the Gatekeeper to discover that an endpoint is alive, they also each serve separate, unique functions.

The lightweight RRQ is intended to prevent a registration with a Gatekeeper from expiring. The message is generated by the endpoint and does not require the Gatekeeper to poll each endpoint on a regular interval. This message is also a means of allowing the Gatekeeper to provide updated registration information, such as a new list of Alternate Gatekeepers, after the initial registration.

Version 1 of H.323 did not have the concept of a lightweight RRQ, so the IRQ/IRR exchange is the only mechanism available to determine endpoint status of Version 1 devices. However, the lightweight RRQ may be a better choice for determining endpoint status for Version 2 and higher devices.

The IRQ/IRR exchange allows the Gatekeeper to poll the endpoint periodically to discover if the endpoint is still alive. However, an IRR is also intended to convey details about current active calls. This can be used by the Gatekeeper to discover calls that have terminated, which may happen

if the endpoint fails to properly send a DRQ message for a call. The IRR message also provides specific details about active calls.

6.10 Specifying the Payload Format for a Channel

Implementers should be conscientious of the fact that there are possibly multiple payload formats defined for media formats. For example, two payload formats are defined for H.263—one is defined for the Recommendation H.263 (1996) and one for Recommendation H.263 (1998). Other payload formats may be defined for existing codecs or revisions of those codecs. For interoperability, it is strongly advised that implementers provide the **mediaPacketization** element of the **h2250LogicalChannelParameters** sequence in the **OpenLogicalChannel** message so that there is no ambiguity at to which payload format is being used.

6.11 Version Dependencies in Annexes

It was noted that the Annexes to H.323 often fail to indicate the minimum version of H.323 and H.245 required for the Annex. This table is an attempt to clarify the version relationships:

H.323 Annex	*Minimum H.323 Version*	*Minimum H.245 Version*
Annex Dv1 (1998)	1998 (Version 2)	1998 (Version 4)
Annex Dv2 (2000)	2000 (Version 4)	2000 (Version 7)
Annex Dv3 (2005)	2000 (Version 4)	2005 (Version 11)
Annex E	1998 (Version 2)	N/A
Annex F	1998 (Version 2)	N/A
Annex G	1998 (Version 2)	1998 (Version 4)
Annex Gv2 (2006)	1998 (Version 2)	2000 (Version 7)
Annex J	1998 (Version 2)	N/A
Annex K	1998 (Version 2)	N/A
Annex L	1998 (Version 2)	N/A
Annex M.1	2000 (Version 4)	N/A
Annex M.2	2000 (Version 4)	N/A
Annex M.3 (2001)	2000 (Version 4)	N/A
Annex M.4 (2004)	2000 (Version 4)	N/A
Annex O	2000 (Version 4)	N/A
Annex P	2000 (Version 4)	2003 (Version 9)
Annex Q	1998 (Version 2)	2000 (Version 7)
Annex R	2000 (Version 4)	N/A

6.12 Routing through Signaling Entities and Detecting Loops

In some call scenarios, a call may be routed though a signaling entity multiple times. For example, a call from Endpoint 1 (EP1) may be routed through Gatekeeper 1 (GK1) and Gatekeeper 2 (GK2) to Endpoint 2 (EP2) as shown in Figure 1.

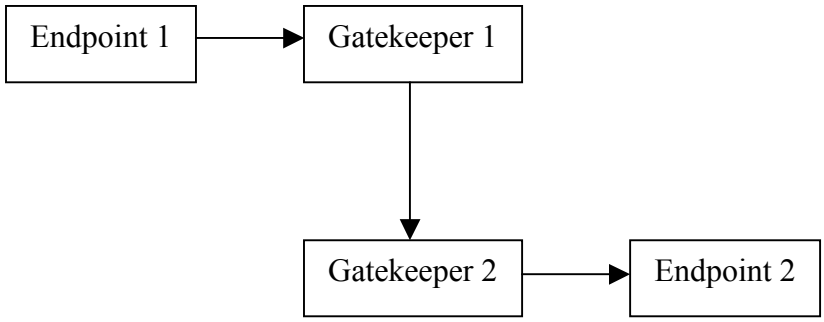


Figure 1 - Call placed through multiple gatekeepers

If EP2 redirects the call to a third endpoint, such as Endpoint 3 (EP3), signaling entities such as GK1 and GK2 should be prepared to handle such call rerouting. For this example, assume that EP2 returned a Facility message with a **reason** of **callForwarded** upon receiving a Setup message. Rather than propagate that response back to EP1, GK2 may choose to handle the call forward operation. GK2 would send a Release Complete to EP2 and begin rerouting the call. Suppose that GK2 sends an LRQ message to GK1 for EP3 and that GK1 replies with its address so that that calls routed to EP3 are routed through it. GK2 would then send a Setup message for this call to GK1 as shown in Figure 2.

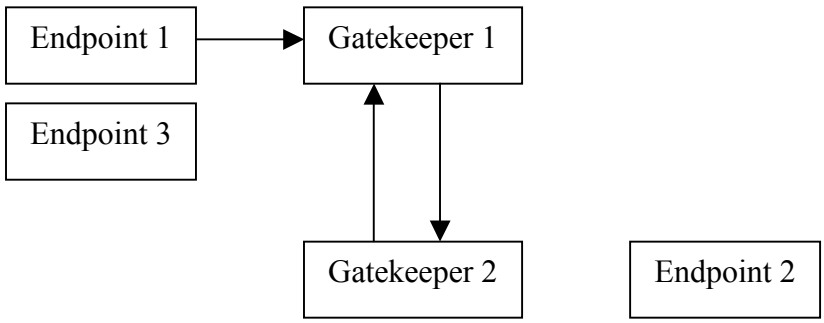


Figure 2 - Gatekeeper 2 re-routes call back to Gatekeeper 1

When GK1 receives the Setup message from GK2, it may inadvertently mistake the call as "bogus", since the Call Identifier will match an already existing call within the Gatekeeper. Implementers should consider this type of call scenario and be prepared to receive incoming calls that contain Call Identifiers for calls that are already being routed through the routing entity. The routing entity should examine not only the Call Identifier, but also the destination address of the call (the call signaling address, aliases, or Called Party Number of the destination). In this case, the call is routed through GK1 with a destination address of EP2 is rerouted by GK2 to GK1, but with a destination address of EP3. In this way, the GK1 will properly handle call routing and rerouting, as well as prevent loops in the call signaling path.

In this example, there was a dependency on the H.323v2 Call Identifier. Unfortunately, H.323 version 1 systems did not have Call Identifiers. For this reason, these loop detection and rerouting procedures are not possible. Nonetheless, it is advisable for routing entities to make an effort to

prevent loops properly. For example, if the entities in Figure 2 were version 1 devices, the GK1 may examine the source address, destination address, and Conference Identifier (CID) of the call. The first time the call is presented to the Gatekeeper, the destination address is EP2, just as before. However, when GK re-routes the call back to GK1, the destination address is EP3. In this way, GK1 may allow proper rerouting of the call to EP3.

The logic for Version 1 devices seems similar to that for Version 2 and higher devices, but there are issues when EP2 and EP3 are MCUs, for example. Suppose that EP2 is an MCU that is directing all calls to EP3. The first time a call is redirected to GK1, GK1 may realize that this is, indeed, a call redirection as described above. However, when the second call is redirected, GK1 has no means of distinguishing between the first redirected call and the second: the source address *may* be the same, the destination address is the same as the previously rerouted call (EP3), and the Conference ID is the same. So in this case, GK1 may have no choice but to assume that a loop has occurred and release the offending call. Although this is unfortunate, H.323v2 and higher systems do not suffer from this problem. What is important, though, is that loop detection is possible—even with version 1 systems.

6.13 Packetization for G.729, G.729a, G.711, and G.723.1

The delay associated with codec processing and packetization should be kept as short as possible. To accomplish this objective when G.729 or G.729A is used, two frames per packet should be considered as the maximum packet size. Similarly, G.711 may be used with packet sizes of 10 ms (80 frames) or 20 ms (160 frames) to achieve this objective. Finally, when G.723.1 is used, only one frame should be included in each packet. The 30 ms frame size of G.723.1 results in speech collection and coding delay of at least 60 ms, contributing to difficulty of interactive communications.

6.14 Checking versions for T.38 and V.150.1

It is important that devices properly negotiate the version of the T.38 or V.150.1 to be used and agree to use the same version. At the present time there are few guidelines for version negotiation. Until the guidelines are developed the following note applies:

Devices supporting multiple versions of T.38 and V.150.1 may offer multiple proposals in Fast Connect, each with a different version specified. A device shall not accept a proposal for a version that it does not support.

7 Allocated Object Identifiers and Port Numbers

Information in this section is provided for informational purposes and convenience. This section does not supercede nor replace proper references in H.225.0, H.225, H.235, or other Recommendations.

7.1 Allocated Object Identifiers

The following object identifiers have been allocated for protocols associated with H.323. Any future object IDs that are allocated should be indexed here to prevent duplication.

Note that object IDs below that are allocated below the arc { itu-t(0) recommendation(0) } are show with an abbreviated prefix of "0 0" below.

{ 0 0 h(8) 2250 version(0) *[v]* }	H225.0 version numbers
Assigned values of *v*: 1-4	
{ 0 0 h(8) 2250 annex(1) g(7) version(0) *[v]* }	H225.0 Annex G version numbers
Assigned values of *v*: 1-2	
{ 0 0 h(8) 2250 annex(1) g(7) usage(1) *[u]* }	H225.0 Annex G usage tags
Assigned values of *u*: none	
{ 0 0 h(8) 245 version(0) *[v]* }	H245 version numbers
Assigned values of *v*: Please refer to Table D.1/H.245	
{ 0 0 h(8) 245 generic-capabilities(1) video(0) *[c]* }	Generic video capabilities
Assigned values of *c*: Please refer to Table D.1/H.245	
{ 0 0 h(8) 245 generic-capabilities(1) audio(1) *[c]* }	Generic audio capabilities
Assigned values of *c*: Please refer to Table D.1/H.245	
{ 0 0 h(8) 245 generic-capabilities(1) data(2) *[c]* }	Generic data capabilities
Assigned values of *c*: Please refer to Table D.1/H.245	
{ 0 0 h(8) 245 generic-capabilities(1) control(3) *[c]* }	Generic control capabilities
Assigned values of *c*: Please refer to Table D.1/H.245	
{ 0 0 h(8) 245 generic-capabilities(1) multiplex(4) *[c]* }	Generic multiplex capabilities
Assigned values of *c*: Please refer to Table D.1/H.245	
{ 0 0 h(8) 283 generic-capabilities(1) 0 }	H.283 Capability
{iso (1) identified-organization (3) icd-ecma (0012) private-isdn-signalling-domain (9)}	Identifies QSIG as the tunneled protocol within an H.225.0 Call Signalling Channel

7.2 Allocated Port Numbers

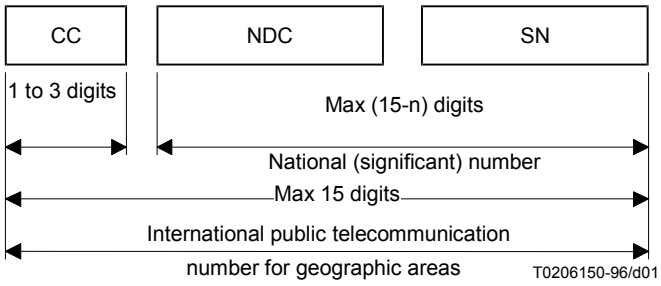
The following IP port numbers have been allocated for various components of H.323:

1300	TLS secured call signalling
1718	Multicast RAS Signalling
1719	Unicast RAS Signalling
1720	TCP call signalling
2099	Annex G/H.225.0 Signalling
2517	Annex E/H.323 Signalling

8 Use of E.164 and ISO/IEC 11571 Numbering Plans

8.1 E.164 numbering plan

ITU-T Recommendation defines E.164 numbers the following way for geographic areas:



CC Country Code for geographic areas
NDC National Destination Code (optional)
SN Subscriber Number
n Number of digits in the country code

NOTE – National and international prefixes are not part of the international public telecommunication number for geographic areas.

Figure – International public telecommunication number structure for geographic areas

Similar descriptions are also defined for non-geographic areas. Recommendation E.164 further defines country codes (CC) for all the countries and regions of the world.

An international E.164 number always starts with a country code and its total length is always 15 digits or less. More importantly, it does not include any prefixes that are part of a dialing plan (for example, "011" for an international call placed in North America, or "1" for a long-distance call), nor does it include "#" or "*". The number "49 30 345 67 00" is an E.164 number with CC=49 for Germany. A national number is the international number stripped of the country code, "30 345 67 00" in this case. The subscriber number is the national number stripped of the national destination code, "345 67 00" in this case.

An E.164 number has global significance: any E.164 number can be reached from any location in the world. A "dialed digit sequence", however, only has significance within a specific domain. Within a typical private numbering plan in an enterprise, for example, a prefix, such as "9", may indicate that a call goes "outside", at which point the local telephone company's dialing plan takes over. Each telephone company or private network is free to choose its own dialing plan. It is also free to change it as it pleases—and frequently does so (adding new area codes, for example).

In a typical geographically determined network where users input telephone numbers manually and where users do not travel too much, having different dialing plans everywhere is usually a problem. However, when a user travels, the user must determine the other network's numbering plan in order to place calls. When computer systems perform the dialing automatically, the user is usually required to customize the dialing software for every region or network.

Because of these issues with varying dialing plans and automated dialing, it is essential to be able to refer to an absolute "telephone number" instead of "what you have to dial to reach it from a specific location." Proper usage of E.164 numbers can resolve these issues. Many systems use E.164 numbers instead of dialed digits: for example, a PBX may gather the dialed digits from a user on a telephone and then initiate a call to the local phone company using an E.164 number in the Called Party Number information element in Q.931. When completing the Called Party Number IE, specifying the numbering plan as "ISDN/telephony numbering plan (Recommendation E.164)" indicates an E.164 number. Specifying the type of number as "unknown" and the specifying the numbering plan as "unknown" indicates dialed digits.

The following are a set of definitions from E.164:

number

A string of decimal digits that uniquely indicates the public network termination point. The number contains the information necessary to route the call to this termination point.

A number can be in a format determined nationally or in an international format. The international format is known as the International Public Telecommunication Number which includes the country code and subsequent digits, but not the international prefix.

numbering plan

A numbering plan specifies the format and structure of the numbers used within that plan. It typically consists of decimal digits segmented into groups in order to identify specific elements used for identification, routing and charging capabilities, e.g. within E.164 to identify countries, national destinations, and subscribers.

A numbering plan does not include prefixes, suffixes, and additional information required to complete a call.

The national numbering plan is the national implementation of the E.164 numbering plan.

dialing plan

A string or combination of decimal digits, symbols, and additional information that define the method by which the numbering plan is used. A dialing plan includes the use of prefixes, suffixes, and additional information, supplemental to the numbering plan, required to complete the call.

address

A string or combination of decimal digits, symbols, and additional information which identifies the specific termination point(s) of a connection in a public network(s) or, where applicable, in interconnected private network(s).

prefix

A prefix is an indicator consisting of one or more digits, that allows the selection of different types of number formats, networks and/or service.

international prefix

A digit or combination of digits used to indicate that the number following is an International Public Telecommunication Number.

country code (CC) for geographic areas

The combination of one, two or three digits identifying a specific country, countries in an integrated numbering plan, or a specific geographic area.

national (significant) number [N(S)N]

That portion of the number that follows the country code for geographic areas. The national (significant) number consists of the National Destination Code (NDC) followed by the Subscriber Number (SN). The function and format of the N(S)N is nationally determined.

national destination code (NDC)

A nationally optional code field, within the E.164 number plan, which combined with the Subscriber's Number (SN) will constitute the national (significant) number of the international public telecommunication number for geographic areas. The NDC will have a network and/or trunk code selection function.

The NDC can be a decimal digit or a combination of decimal digits (not including any prefix) identifying a numbering area within a country (or group of countries included in one integrated numbering plan or a specific geographic area) and/or network/services.

national (trunk) prefix

A digit or combination of digits used by a calling subscriber, making a call to a subscriber in his own country but outside his own numbering area. It provides access to the automatic outgoing trunk equipment.

subscriber number (SN)

The number identifying a subscriber in a network or numbering area.

8.2 Private Network Number

Private Network Numbers are used in private or virtual private telephony networks, e.g., a corporate network of PBXs and virtual private lines.

ISO/IEC 11571 defines Private Network Number (PNP) as having up to three regional levels.

A PNP Number shall comprise a sequence of x decimal digits (0,1,2,3,4,5,6,7,8,9) with the possibility that different PNP Numbers within the same PNP can have different values of x. The maximum value of x shall be the same as for the public ISDN numbering plan, see ITU-T Recommendation E.164.

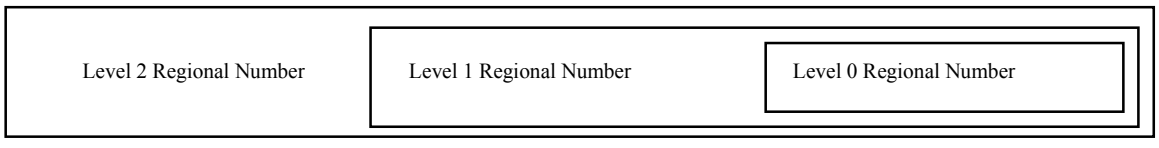


Figure – H.323 - Structure of a PNP Number with three levels of regions

A level n Regional Number (RN) shall have significance only within the level n region to which it applies. When that number is used outside that level n region, it shall be in the form of an RN of level greater than n. Only a Complete Number shall have significance throughout the entire PNP.

A typical example in North America would be a 4-digit "extension" as the Level 0 Regional Number: a 3-digit "location code" combined with the 4 digit "extension" would form the Level 1 Regional Number. The Level 2 Regional Number would be nil.

A prefix could also be used to signal which regional number is used, and would not be part of the regional number per se, but only part of the dialing plan. Again, a typical example would be the use of digit "6" to access a Level 1 Regional Number, and no digit for a Level 0 Regional Number.

The following are a set of definitions from ISO/IEC 11571:

Private Numbering Plan (PNP)

The numbering plan explicitly relating to a particular private numbering domain, defined by the PISN Administrator of that domain.

PNP Number

A number belonging to a PNP.

Region

The entire domain or a sub-domain of a PNP. A region does not necessarily correspond to a geographical area of a PISN.

Region Code (RC)

The leading digits of a PNP Number which identify a region. The RC may be omitted to yield a shortened form of a PNP Number for use internally to that region.

Regional Number (RN)

A particular form of a PNP Number which is unambiguous in the region concerned.

Complete Number

A number which is unambiguous in the entire PNP, i.e. which corresponds to the highest regional level employed in that PISN.

9 ASN.1 Usage, Guidelines, and Conventions

9.1 NULL, BOOLEAN, and NULL/BOOLEAN OPTIONAL

Throughout the ASN.1 used in H.323-series documents, the reader will see the types NULL and BOOLEAN used, along with the modifier OPTIONAL in some cases. People have questioned when NULL should be used or when BOOLEAN should be used and what the semantic differences are.

The BOOLEAN type allows a TRUE or FALSE value to be conveyed in the protocol. When used in conjunction with OPTIONAL, it actually allows three values to be conveyed through the protocol: TRUE, FALSE, and *absent*. The question is what does *absent* mean? In some instances, the absence of a BOOLEAN OPTIONAL means should be interpreted as FALSE, while in other cases, it should be interpreted as "I don't care" or "I don't know"—but not always. For example, the **additiveRegistration** field in the RRQ of H.225.0 Version 4 is defined as a BOOLEAN OPTIONAL. When present, it clearly indicates that the endpoint supports the feature or does not support the

feature. However, absence of this field shall also be interpreted as FALSE. The reason is that an older endpoint would not know anything about the field and would obviously not be able to include it. Moreover, they certainly do not support the feature. Another example is the **originator** field in the **perCallInfo** sequence. When present, the meaning is quite clear: the caller is the originator or the terminator of the call. However, if the field is not present, it may mean that the endpoint does not know or cannot supply this information for some reason.

The NULL type is often used to select one of several CHOICE options. NULL carries no particular value, as it merely indicates presence. In selecting the conference goal in a Setup message, for example, the goal CHOICEes are simply NULL types to allow the endpoint to indicate a selection. Another common use of NULL is with the OPTIONAL modifier. A NULL OPTIONAL type allows an endpoint to indicate support for a feature, for example. It is similar in semantics to a BOOLEAN in that the presence of a NULL field indicates TRUE and absence of the NULL field indicates a FALSE. As an example, the **fastConnectRefused** field in the Alerting message is a NULL OPTIONAL. Absence of the field is interpreted as FALSE—Fast Connect is not (yet) refused. Presence of the field, though, clearly indicates refusal of Fast Connect. So why was BOOLEAN not used as the type for this field? It would not have made the encoding any clearer, because the field is past the extension marker (ellipsis). A version 1 and 2 device, for example, would not know to send this field, so there would be three values to consider if BOOLEAN were used: TRUE, FALSE, and *absent*.

Ideally, a field will convey no more values than makes sense. In most cases, these types indicate only two possible values: TRUE/present or FALSE/absent. However, there may be cases where three values are intended and the reader should refer to the appropriate Recommendation to determine if, indeed, there is significance in tri-state fields.

9.2 ASN.1 Usage in H.450-Series Recommendations

This section summarizes the use of ASN.1 in the current H.450.x Recommendations. This information is provided for implementers of the H.450.x protocols, as well as authors of new H.450.x Recommendations.

9.2.1 ASN.1 version and encoding rules

The ASN.1 code in H.450.x is based on the 1994 version of X.680-683, including the amendments on "*Rules of extensibility*".

The *basic aligned variant* of *packed encoding rules* (PER) is used as specified in X.691 (1995).

9.2.2 Tagging

All modules defined in Recommendations H.450.x use the *tag default* AUTOMATIC TAGS.

The ROS APDUs (see below) are defined in H.450.1 as *tagged types* within the CHOICE type ROS. No other type defined in H.450.x is a *tagged type*, i.e. all *sets, sequences* and *choices* (except ROS) are automatically tagged.

9.2.3 Basic ASN.1 Types

The following types occur in ASN.1 definitions of H.450.x:

BMPString, NumericString	NULL
BOOLEAN	OBJECT IDENTIFIER
CHOICE	OCTET STRING
CLASS (see below)	*Open type (see below)*
ENUMERATED	SEQUENCE
GeneralizedTime	SEQUENCE OF
INTEGER	SET OF

No use is currently foreseen for the following basic types (needs consideration on a case-by-case basis):

CHARACTER STRING	ObjectDescriptor
EMBEDDED PDV	REAL
EXTERNAL	UTCTime
GeneralString, GraphicString, PrintableString, TeletexString (T61String), UniversalString, VideotexString, VisibleString (ISO646String)	

Use of the following basic types in future Recommendations H.450.x should not be precluded (needs consideration on a case-by-case basis):

BIT STRING	Selection Type (out of a CHOICE)
IA5String	SET
INSTANCE OF	TYPE-IDENTIFIER (see X.681)

Note: Some of these types are already used by other Recommendations in the H.323 universe, e.g. BIT STRING and TYPE-IDENTIFIER in H.235.

9.2.4 Value sets, subtyping and constraints used in H.450.x:

H.450.x Recommendations use *size constraints* (strings, set-of and sequence-of) and *value range* constraints (integers). In H.450.1 *inner subtyping* ("WITH COMPONENTS") is used occasionally.

The use of *value sets*, *single values*, *contained subtypes* and *permitted alphabets* should be possible if needed by future services. The *type constraint* (for restricting an *open type*) may be useful, too.

Explicit set arithmetic (UNION, INTERSECTION, EXCEPT, ALL EXCEPT) is currently not used on subtype specifications.

9.2.5 Object classes, parameterization, general constraints, and ROS

H.450.1 defines a *remote operations service* (ROS) based on X.880. ROS uses *object classes* (X.681), *parameterization* (X.683) and *constraints* (X.682) for its generic part.

Two object classes OPERATION and ERROR are defined and then used to define four PDU types (*Invoke, ReturnResult, ReturnError* and *Reject*) as sequences containing individual parts of these classes. The first three PDU types contain an optional *open type* component which is tied by a *table constraint* ("at (@)" notation) to the code value identifying the particular operation or error.

For each supplementary service the actual operations and errors are then defined as *object instances* of the generic classes OPERATION and ERROR in the corresponding Rec. H.450.x. Each operation and error is identified uniquely (within the context of the H.450.x series) by a code value (type INTEGER). A list of currently assigned operation and error values is contained in section 10.8 below.

Each supplementary service defines an *object set* containing all operations defined for that service.

9.2.6 Extensibility and non-standard information

Wherever meaningful, an *extension marker* (ellipsis "...") is included in the definitions.

All operations, and some errors, include placeholders for non-standard (e.g. manufacturer-specific) information. This non-standard information can either be of type *NonStandardParameter* (imported from H.225.0) or of type *Extension*, which is defined in H.450.1 and consists of an *object identifier* followed by an *open type*. The definition of the Extension type uses an *object class* (EXTENSION) with *parameterization* and *constraints* similar to the ROS definition.

Usually there is space for more than one addition of non-standard information in an operation. Additions of both types (NonStandardParameter and Extension) can be mixed in any order.

9.2.7 List of Operation and Error Codes

Table 10.1: ASN.1 Operation values used in H.450 series

Value number	Value name	Defined in standard:
0	callingName	H.450.8
1	~~called~~alertingName	H.450.8
2	connectedName	H.450.8
3	busyName	H.450.8
7	~~callTransferIdentity~~callTransferIdentify	H.450.2

8	callTransferAbandon	H.450.2
9	callTransferInitiate	H.450.2
10	callTransferSetup	H.450.2
11	callTransferActive	H.450.2
12	callTransferComplete	H.450.2
13	callTransferUpdate	H.450.2
14	subaddressTransfer	H.450.2
15	activateDiversionQ	H.450.3
16	deactivateDiversionQ	H.450.3
17	interrogateDiversionQ	H.450.3
18	checkRestriction	H.450.3
19	callRerouting	H.450.3
20	divertingLegInformation1	H.450.3
21	divertingLegInformation2	H.450.3
22	divertingLegInformation3	H.450.3
23	cfnrDivertedLegFailed	H.450.3
27	ccnrRequest	~~Draft~~ H.450.9
28	ccCancel	~~Draft~~ H.450.9
29	ccExecPossible	~~Draft~~ H.450.9
31	ccRingout	~~Draft~~ H.450.9
32	ccSuspend	~~Draft~~ H.450.9
33	ccResume	~~Draft~~ H.450.9
34	callOfferRequest	H.450.10
40	ccbsRequest	~~Draft~~ H.450.9
43	callIntrusionRequest	H.450.11
44	callIntrusionGetCIPL	H.450.11
45	callIntrusionIsolate	H.450.11
46	callIntrusionForcedRelease	H.450.11
47	callIntrusionWOBRequest	H.450.11
49	cfbOverride	H.450.10 (re-used in H.450.11)
80	mwiActivate	H.450.7
81	mwiDeactivate	H.450.7
82	mwiInterrogate	H.450.7

84	cmnRequest	H.450.12
85	cmnInform	H.450.12
100	divertingLegInformation4	H.450.3
101	holdNotific	H.450.4
102	retrieveNotific	H.450.4
103	remoteHold	H.450.4
104	remoteRetrieve	H.450.4
105	callWaiting	H.450.6 (re-used in H.450.10, H.450.11)
106	cpRequest	H.450.5
107	cpSetup	H.450.5
108	groupIndicationOn	H.450.5
109	groupIndicationOff	H.450.5
110	pickrequ	H.450.5
111	pickup	H.450.5
112	pickExe	H.450.5
113	cpNotify	H.450.5
114	cpickupNotify	H.450.5
115	remoteUserAlerting	H.450.10 (re-used in H.450.11)
116	callIntrusionSilentMonitor	H.450.11
117	callIntrusionNotification	H.450.11

Table 10.2: ASN.1 Error Values used in H.450 series

Value number	Value name	Defined in standard:
0	userNotSubscribed	H.450.1
1	rejectedByNetwork	H.450.1
2	rejectedByUser	H.450.1
3	notAvailable	H.450.1
5	insufficiantInformation	H.450.1
6	invalidServedUserNumber	H.450.1
7	invalidCallState	H.450.1
8	basicServiceNotProvided	H.450.1
9	notIncomingCall	H.450.1

10	supplementaryServiceInteractionNotAllowed	H.450.1
11	resourceUnavailable	H.450.1
12	invalidDivertedNumber	H.450.3
14	specialServiceNumber	H.450.3
15	diversionToServedUserNumber	H.450.3
24	numberOfDiversionsExceeded	H.450.3
25	callFailure	H.450.1
31	notActivated	H.450.7
43	proceduralError	H.450.1
1000	temporarilyUnavailable	H.450.3, H.450.11
1004	invalidReroutingNumber	H.450.2
1005	unrecognizedCallIdentity	H.450.2
1006	establishmentFailure	H.450.2
1007	notAuthorized	H.450.3, H.450.11
1008	unspecified	H.450.2, H.450.3
1009	notBusy	H.450.11
1010	shortTermRejection	~~Draft~~ H.450.9
1011	longTermRejection	~~Draft~~ H.450.9
1012	remoteUserBusyAgain	~~Draft~~ H.450.9
1013	failureToMatch	~~Draft~~ H.450.9
1018	invalidMsgCentreId	H.450.7
2000	callPickupIdUnvalid	H.450.5
2001	callAlreadyPickedUp	H.450.5
2002	undefined	H.450.4, H.450.5, H.450.7, (re-used in H.450.9, H.450.11, H.450.12)

Annex: Defect Report Form for Recommendations of the H.323 System

DATE:	
CONTACT INFORMATION **NAME:** **COMPANY:** **ADDRESS:** **TEL:** **FAX:** **EMAIL:**	
AFFECTED RECOMMENDATIONS:	
DESCRIPTION OF PROBLEM:	
SUGGESTIONS FOR RESOLUTION:	

NOTE - Attach additional pages if more space is required than is provided above.